НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено:
Завідувач кафедри
_____ Сергій СТІРЕНКО
«___»_____20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Комп'ютерні системи та мережі»

спеціальності 123 «Комп'ютерна інженерія»

на тему: «Пристрій стеження за станом пацієнта»

Виконав:
студент IV курсу, групи ІО-63
Сластін Олексій Юрійович _____

Керівник:
проф. д. т. н.
Сімоненко Валерій Павлович _____

Консультант нормоконтроль:
проф. д. т. н.
Сімоненко Валерій Павлович _____

Рецензент:

_____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 123 «Комп'ютерна інженерія»

Освітньо-професійна програма «Комп'ютерні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ СЕРГІЙ СТІРЕНКО
«___»_____20_ р.

ЗАВДАННЯ
на дипломний проєкт студенту
Сластіну Олексію Юрійовичу

1. Тема проєкту «Пристрій стеження за станом пацієнта», керівник проєкту Сімоненко Валерій Павлович, проф., д.т.н., затверджені наказом по університету від «07» травня 2020 р. №1081-С

2. Термін подання студентом проєкту _____

3. Вихідні дані до проєкту *Технічна документація. Теоретичні та статистичні дані. Мікроконтролерна система NodeMCU V3 ESP8266 та Arduino Nano. Датчики: температури DS18B20, сердцевого ритму Pulse Sensor. Середовище розробки Arduino IDE, C++.*

4. Зміст пояснювальної записки *Аналіз предметної області, дослідження методики побудови пристрою на базі мікроконтролерної системи NodeMCU V3, розробка пристрою стеження за станом пацієнта.*

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо) блок-схема алгоритму, функціональна схема, структурна схема сценарію використання

6. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	*Затвердження теми роботи*	*01.09.2019*	
2	*Вивчення та аналіз завдання*	*30.03.2020*	
3	*Розробка архітектури та загальної структури систем*	*10.04.2020*	
4	*Розробка структур окремих підсистем*	*22.04.2020*	
5	*Програмна реалізація системи*	*04.05.2020*	
6	*Оформлення пояснювальної записки*	*17.05.2020*	
7	*Передзахист*	*26.05.2020*	
8	*Захист*	*16.06.2020*	

Студент Олексій СЛАСТІН

Керівник Валерій СІМОНЕНКО

Анотація

Дана бакалаврська робота присвячена розробці пристрою, що реєструє пульс пацієнта, а також температуру його тіла, з подальшим відправленням цієї інформації на Google Диск де її можна переглянути за допомогою ПК, ноутбука чи смартфона. Ця система може попередити пацієнтів до того, як стан їх здоров'я критично погіршиться. Окрім того дані отримані мікроконтролером можна зберігати для подальшої їх перевірки або обробки згодом.

Annotation

This bachelor's thesis is dedicated to development of the device which register the pulse of a patient and also the temperature of his/her body with further transffering it on the Google disk,where it can be watched with the help of PC,laptop or smartphone.This system can warn patients before their health condition become critically worse.Besides,the database which was obtained by micro conroller, can save for their further review or working-up.

ТЕХНІЧНЕ ЗАВДАННЯ

до дипломного проєкту
освітньо-кваліфікаційного рівня бакалавр

на тему: “Пристрій стеження за станом пацієнта”

Київ – 2020 року

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Найменування: «Пристрій стеження за станом пацієнта».

Область застосування мого пристрою: альтернатива існуючим методам стеження за станом пацієнта у медицині, також у спортивній сфері чи у побуті.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки цього пристрою є завдання на виконання дипломного проекту, пристрою стеження за станом пацієсента, затвердженого кафедрою обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського».

3. МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою мого проекту є розробка пристрою, що має вимірювати пульс та температуру тіла людини, та у режимі реального часу надсилати отримані дані через бездротову мережу WI-FI на Google Диск.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелами розробки мого проекту служать науково-технічна література, довідники по мікроконтролерам, публікації в спеціалізованих виданнях, та публікації на цю тему в мережі Інтернет.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до продукту, що розробляється

- Визначення температури тіла за допомогою датчика;
- Вимірювання пульсу людини у режимі реального часу за домогою датчика;
- Можливість покращити точність вимірювання при використанні більш досконалих датчиків;

5.2. Вимоги до програмного забезпечення

- Операційна система Linux, macOS, Windows або Android чи iOS для смартфонів ;
- Доступ до мережі Інтернет.
- Браузер Google Chrome, Firefox, Safari або Microsoft Edge;

5.3. Вимоги до апаратної частини

- Джерело живлення 5 – 10 В.
- Необхідне підключення до мережі WI-FI;

6. ЕТАПИ РОЗРОБКИ

	Дата
Вивчення літератури	30.01.2020
Складання і узгодження технічного завдання	11.02.2020
Створення модулів системи, що розробляється	14.03.2020
Тестування окремих модулів системи	10.04.2020
Доопрацювання, налагодження і виправлення помилок	04.05.2020
Оформлення документації дипломної роботи	17.05.2020

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЄКТУ

до дипломної роботи
освітньо-кваліфікаційного рівня__бакалавр___

на тему: “Пристрій стеження за станом пацієнта”

Київ – 2020 року

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЄКТУ

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проект	2	
2	А4	ДП.467100.001 ТЗ	Технічне завдання	4	
3	А4	ДП.467100.002 ВП	Відомість проєкту	1	
4	А4	ДП.467100.003 ПЗ	Пояснювальна записка	60	
5	А4	ДП.467100.004 Д1	Схема структурна	1	
6	А4	ДП.467100.005 Д2	Блок-схема алгоритму	1	
7	А4	ДП.467100.006 Д3	Схема функціональна	1	

					ІАЛЦ.467100.002 ВП		
Змн.	*Арк.*	*№ докум.*	*Підпис*	*Дата*			
Розроб.		*Сластін О.Ю..*			*Відомість дипломного проекту*	*Літ.*	*Арк.* 2 / *Акрушів* 1
Перевір.		*Сімоненко В.П.*					
Н. Контр.		*Сімоненко В.П.*				*НТУУ «КПІ ім. Ігоря Сікорського», ФІОТ ІО-63*	
Затверд.							

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проєкту

на тему: “Система стеження за станом пацієнта ”

Київ – 2020 року

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

GPIO – general-purpose input/output

IDE – integrated drive electronics

GND – ground

UDP – user datagram protocol

HTTPS – hyper text transfer protocol secure

URL – uniform resource locator

NTP – network time protocol

TLS – transport layer security

ВСТУП

Частота серцевих скорочень та температура тіла є одними із найважливіших показників стану здоров’я людини. На відміну від рентгену, який є шкідливим для здоров'я, вони мають перевагу у легкому доступі та діагностуванні,. Частота серцевих скорочень – це фізіологічний показник нормального ритму серцебиття людини, цей показник використовується у медичній та спортивній сфері, або ж просто у побуті. Нормальною (середньою) частотою серцевих скорочень у стані спокою вважається приблизно 60-80 ударів за хвилину. Слід зазначити, що частота серцевих скорочень залежить від віку людини, а також її статі та можливих захворювань. У жінок частота серцевих скорочень на 5-10 ударів менша ніж у чоловіків.

Однією з найважливіших проблем громадськості є здоров’я людини. Всі інші речі стають безглуздими та непотрібними, якщо хтось захворіє або помре. Саме з цієї причини люди витрачають багато грошей задля збереження власного здоров’я. Проте вони чомусь завжди вважають, що отримувати серйозну медичну допомогу вже надто пізно. Але це не так, якщо вчасно звернутись до медичного закладу, то можна діагностувати захворювання у людини та надати кваліфіковану медичну допомогу. Однак у зв’язку зі скрутним фінансовим положенням людини, або ж великим віддаленням медичних установ від місця проживання, звернення до них стає якщо не неможливим, то дуже незручним та дорогим. У такій ситуації можна використовувати розроблений мною пристрій. Він простий у використанні, тож людина самостійно може виміряти показники серцевих скорочень та температуру свого тіла, а вони в свою чергу завантажаться на Google Диск, де їх може переглянути кваліфікований лікар. На Google Диску дані представлені у вже обробленому, структурованому вигляді, тож лікар не повинен буде

витрачати час на їх розшифровування та складний аналіз. Це один з можливих прикладів використання мого проекту.

Інша ситуація коли люди вже знаходяться у лікарні та потребують нагляду. Через це біля них чергують медсестри які час від часу перевіряють стан пацієнтів. Мій пристрій може пришвидшити та оптимізувати їх роботу, оскільки життєві показники пацієнтів одразу будуть надходити до комп'ютера чергової медсестри. Доступ до них зможе отримати лікар, якому знадобилася медична карта пацієнта, і переглянути виміри за весь час нагляду за пацієнтом, а не тільки ті, які він щойно зняв.

Також пристрій можуть використовувати спортсмени, щоб переглядати свій пульс під час тренувань чи після них, або звичайні люди для буденного стеження за показниками організму.

Перевагами мого пристрою є його простота, користуватися ним зможе будь-яка людина. А також легкість його побудови, за необхідності можна додати чи замінити якийсь з датчиків.

Враховуючи усе вище сказане, була поставлена така задача: розробити пристрій, який міг би вимірювати частоту серцевих скорочень і температуру тіла та відправляти ці показники на Google Диск.

РОЗДІЛ 1
ОГЛЯД АНАЛОГІВ

1.1Аналіз готових рішень

Ринок «розумних» медичних пристроїв швидко розвивається, та з кожним роком стає набагато більшим за обсягом. Виробники коли розробляють нові пристрої намагаються додати в них якомога більше додаткових функцій, що в свою чергу призводить до виробництва багатофункціональних пристроїв. У зв’язку з таким швидким прогресом стає дедалі складніше віднести ці девайси до якоїсь певної категорії. За допомогою них користувач може не тільки проводити моніторинг власного стану здоров’я, а й відправляти ці дані лікареві. Повсюдне поширення Інтернет Речей передбачає підключення медичних приладів до мережі Інтернет, задля відправки даних моніторингу на хмарні сервіси. У цьому розділі я навів приклади таких «розумних» медичних пристроїв.

Фітнес-трекери

Фітнес-трекер являє собою гаджет компактного розміру, що вдягається на руку та призначений для контролю фізичної активності людини. В даний час виробництвом таких пристроїв займається велика кількість виробників електроніки такі як: Jawbone, Garmin, Fitbit, Misfit і інші.

Основним завданням фітнес-трекера є мотивація свого власника на активну діяльність і контроль одержуваного навантаження. Для виконання поставлених завдань в подібних пристроях встановлені датчики, які фіксують дані пульсу, кількості пройдених кроків, витрачених калорій, рівня стресу, якості сну, швидкості переміщення і довжини пройденої відстані. Отримана інформація може передаватися на смартфон або комп'ютер, а використання спеціально розробленого додатка дозволить зробити розрахунки активності

людини, зміни показників здоров'я і при необхідності дасть рекомендації для успішного досягнення поставленої користувачем мети. Найбільш поширеною формою фітнес-трекерів є браслет. Але існують моделі у вигляді кліпс, окулярів і навіть навушників. Існують трекери, здатні самі розпізнати, що людина лягла спати й ті, яким потрібно про це повідомляти, натиснувши відповідну кнопку. Найсучасніші «розумні» трекери вміють визначати в якій, глибокій або легкої фазі сну знаходиться людина і коригують спрацьовування будильника саме на легку фазу, коли людину легше розбудити. [1]

До фітнес-трекерів можна віднести Fitbit Charge 2.

Рис.1.1 Fitbit Charge 2

Цей браслет володіє широким функціоналом, а саме:

Він може відстежувати добову активність людини. Трекер рахує кількість пройдених кроків, спалених калорій та активних хвилин. Ви можете вимірювати пройдену дистанцію та переглядати статистику за минулу добу

або цілий тиждень. Також браслет буде нагадувати про необхідність рухатися, для того щоб допомогти вам залишатися активним протягом всього дня, він буде сповіщати вас та просити робити 250 кроків кожну годину.

Рис.1.2. Датчики

Фітнес-браслет також має велику кількість вбудованих датчиків які можуть фіксувати стан людини. Він може контролювати ваш сон, слідкувати за його якістю та тривалістю. Цим він може допомогти виявити у вас деякі розлади, якщо вони є, та виробити правильні звички, що значно покращить самопочуття.

Трекер вимірює пульс та додаткові показники. Завдяки цьому ви можете дізнатися не тільки частоту серцевого ритму але й максимальну кількість кисню, яку ваше тіло може використовувати під час сильних навантажень. Якщо значення пульсу будуть виходити за допустимі межі браслет попередить вас про небезпеку, а деякі моделі навіть зможуть викликати вам швидку допомогу.

Для керування браслетом та відображення всіх показників, розроблений спеціальній додаток, що встановлюється на смартфон. У ньому можна не тільки переглянути дані, а й побудувати зрозумілі графіки, для наочного бачення результатів. [2]

«Розумний» одяг

Окрім фітнес-трекерів, існують й інші пристрої які можуть відслідковувати основні життєві показники людини. Саме таку функцію й виконує «розумний» одяг. Це одяг який поєднано з сучасними інформаційними технологіями. Зараз «розумний» одяг знаходить широке розповсюдження у сферах медицини, спорту або навіть у військовій.

Цей аксесуар може допомогти людям, що мають фізіологічні вади або захворювання, також допомагає стежити за станом людей, що працюють з небезпечними речовинами. Такий одяг дозволяє проводити контроль основних життєвих показників таких як частота пульсу, частота дихання, температура тіла та інші. Проводити віддалено аналізи та здійснювати дистанційні медичні консультації.

До того ж «розумний» одяг може допомогти людині вижити в екстремальних умовах. За його допомогою можна відслідковувати стан людини та її місцезнаходження, проводити контроль рівня втоми водія або пілота літака.

WEMU

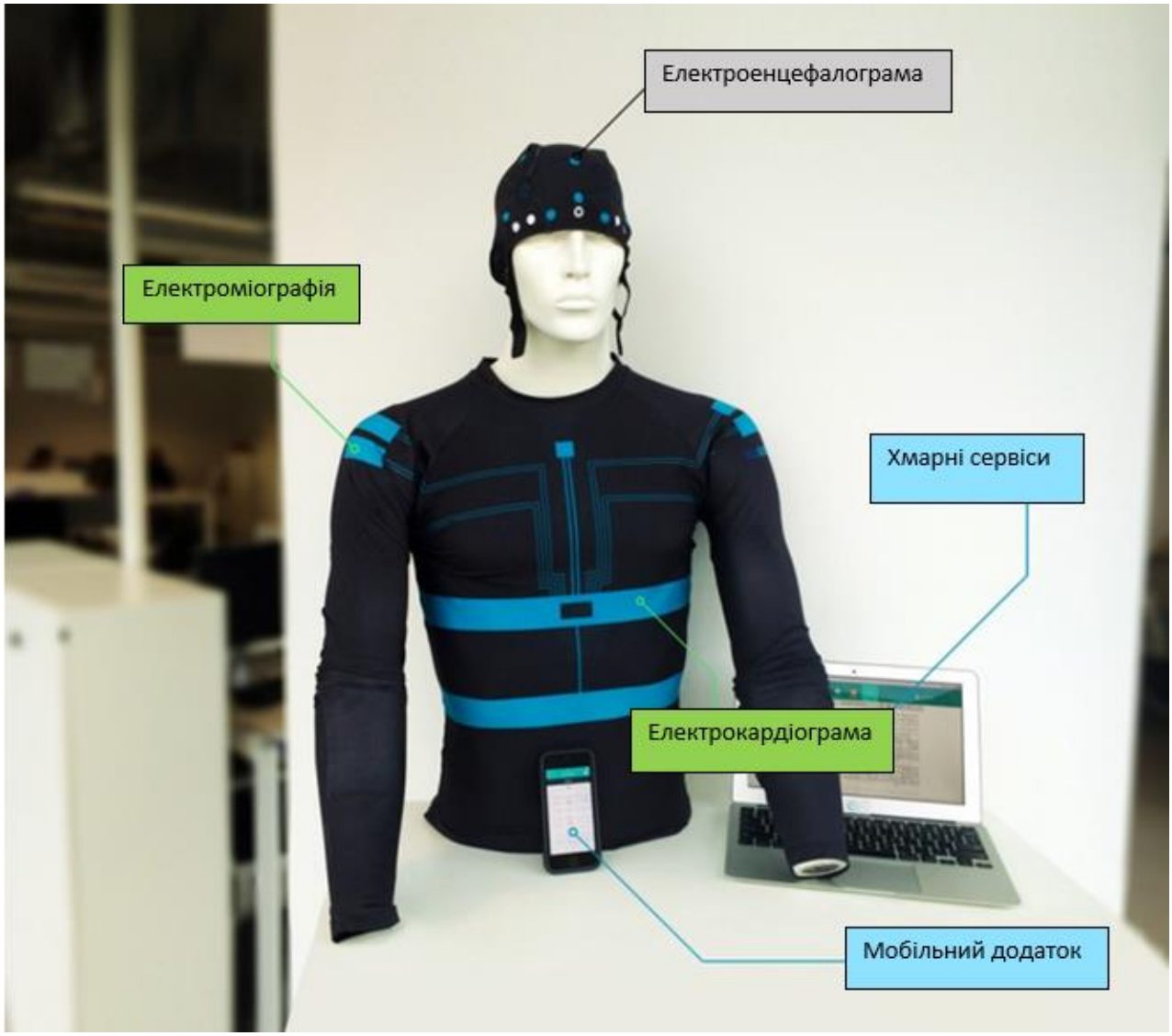


Рис.1.3.

WEMU – це ще один приклад медичного «розумного одягу». Він створений для хворих епілепсією. Зазвичай моніторинг стану подібних хворих здійснюється в спеціалізованих лікарнях, обладнаних стаціонарними ЕКГ - системами.

Розробники BioSerenity разом з французькими організаціями по дослідженню епілепсії пропонують використовувати замість цього комплект «розумного» одягу, у який входить кофта та шапка з вбудованими датчиками

ЕКГ, ЕЕГ та ЕМГ. Подібна система дозволяє записувати параметри припадків, а також знімати показники активності мозку декілька разів на день. У деяких випадках розробити правильний курс, що підійде пацієнту без подібної інформації практично неможливо.

Wemu використовує запатентовану технологію сухих датчиків в поєднанні з гнучкими електронними схемами, вбудованими в тканину, для того щоб створити зручну для носіння повсякденну систему моніторингу. Прототип такого одягу був розроблений за участі лікарів, як спеціалістів так й лікарів загального профілю, в декількох країнах. Консультативна рада яка складається з трьох лікарів і на далі стежить за розвитком проекту.

Потрібно додати, що інформація не тільки зберігається локально в пам’яті самого пристрою, але й якщо пацієнт дасть свою згоду, завантажується в спеціальний хмарний сервіс, до якого має доступ лише лікар хворого та спеціальні дослідницькі організації які допомагають розробляти індивідуальний курс лікування. [3]

ВИСНОВКИ ДО РОЗДІЛУ 1

Моніторинг стану здоров’я є важливою складовою для його збереження. Застосування такої системи моніторингу дозволить швидко отримувати необхідні дані та вчасно реагувати на відхилення.

У цьому розділі були розглянуті такі системи та наведені їх приклади. Всі вони володіють широким функціоналом та можуть вирішувати багато поставлених перед ними завдань. Але через це вони коштують непомірно дорого, та є неможливими у покупці для великого прошарку населення. Також вони є складними у використанні, що значно зменшує категорію людей які б могли користуватися таким девайсом.

Саме тому є актуальною розробка пристрою який повинен бути дешевим та простим у виробництві, що дозволило би значно знизити його вартість. Ця система в першу чергу має бути простою. Це дозволить легко замінювати в ній деталі у разі несправності або якщо виникне необхідність додати нові датчики. Також буде зрозумілою у використанні для літніх людей.

РОЗДІЛ 2
ВИБІР ПЛАТИ, ДАТЧИКІВ ТА СЕРЕДОВИЩА РОЗРОБКИ

2.1. Опис завдання

Завдання моєї бакалаврської роботи полягає у виготовленні девайсу, що має на меті вести моніторинг основних показників здоров'я людини. Він повинен міряти пульс та температуру тіла людини. Основу пристрою складає плата Arduino Nano, яка буде контролювати роботу датчиків та передавати цю інформацію на плату NodeMCU, що створена на основі модулю ESP8266. Датчики які будуть підключені до Arduino Nano це цифровий датчик температури DS18B20, та імпульсний датчик серцевого ритму Pulse Sensor. За допомогою вбудованого у плату NodeMCU WI-FI модулю, вона передає дані з датчиків на Google Диск, у режимі реального часу, де можна переглядати всі показники, та бачити у який проміжок часу вони були зроблені

2.2. Що таке Arduino

Arduino – це електрона платформа, що має відкритий початковий код та базується на простому у використанні апаратному та програмному забезпеченні. Плати Arduino можуть зчитувати вхідні дані, такі як світло на сенсорі, повідомлення в Twitter або натиснуту кнопку та перетворювати це у вихідний сигнал, вмикати світлодіод, активувати мотор, керувати «розумним» домом, або публікувати щось в інтернеті. Ви можете відправити на мікроконтролер на своїй платі Arduino набір інструкцій, для цього використовується мова програмування Arduino (на базі Wiring) та програмне забезпечення Arduino (IDE), що базується на Processing. Протягом багатьох років контролер Arduino був мозком тисяч проектів, від повсякденних предметів до складних наукових інструментів. Довкола цієї платформи з

відкритим початковим кодом створилась світова спільнота творців – студентів, поціновувачів, художників, програмістів та професіоналів, які внесли свій вклад у неймовірну кількість доступних знань які можуть стати у нагоді та надати допомогу як початківцям так й експертам.

Arduino була розроблена в Ivrea Interaction Design Institute як найпростіший елемент для швидкого створення прототипів, призначених для студентів які не мають досвіду в електроніці та програмуванні. Як тільки вона охопила більш широкий прошарок людей, плата Arduino стала змінюватись для того, щоб адаптуватися до нових потреб та завдань, диференціюя свою пропозицію від звичайних 8-розрядних плат до продуктів для додатків IoT, носимих пристроїв, пристроїв 3D друку та вбудованих середовищ розробки. Усі плати Arduino повністю відкриті, що дає можливість користувачам самостійно їх створювати та згодом адаптувати їх до своїх конкретних потреб. Програмне забезпечення також з відкритим початковим кодом, та воно розвивається завдяки внеску користувачів з усього світу. [4]

2.3 Чому Arduino

Завдяки простому та доступному користувацькому інтерфейсу Arduino використовується у тисячах різних проектів та додатків. Програмне забезпечення Arduino просто у використанні для початківців, але досить гнучке для користувачів з досвідом. Він працює на Mac, Windows, Linux. Викладачі та студенти використовують його для створення недорогих наукових інструментів, для доказів принципів роботи хімії та фізики або для початку програмування та робототехніки. Дизайнери та архітектори створюють інтерактивні прототипи, музиканти та художники використовують їх для інсталяцій та експериментів з новими музичними інструментами. Творці, звісно ж, використовують його для створення багатьох проектів, що представлені наприклад на Maker Faire. Arduino є ключовим інструментом для

вивчення нових речей. Будь-хто: дитина, любитель, художник, програміст – можуть почати майструвати, просто слідкуючи за покроковими інструкціями набору або поділитися ідеями в Інтернеті з іншими членами спільноти Arduino.

Є багато інших мікроконтролерів та платформ мікроконтролерів, доступних для фізичних обчислень. Parallax Basic Stamp, Netmedia BX-24, Phidges, Handyboard MIT та багато інших пропонують аналогічну функціональність. Всі ці інструменти беруть брудні деталі програмування мікроконтролера та запаковують його в простий для використання пакет. Arduino також спрощує процес роботи з мікроконтролерами, але пропонує студентам та любителям деяких переваг перед іншими системами:

- **Недорогі –** Плати Arduino досить недорогі порівняно з іншими платформами мікроконтролерів. Найдешевша версія модулю Arduino може бути зібрана вручну, навіть попередньо зібрані модулі Arduino коштують менше 50 доларів.
- **Багатоплатформність –** Програмне забезпечення Arduino працює на операційних системах Windows, Mac OS, Linux. Більшість інших мікроконтролерних систем обмежені операційною системою Windows.
- **Просте та зрозуміле середовище розробки –** Arduino Software проста у використанні для новачків але достатньо гнучка для професійних користувачів для того, щоб вони теж могли скористатися її перевагами. Для вчителів вона зручно основана на середовищі програмування Processing, тому ті хто навчаються програмуванню у цьому середовищі будуть знайомі з роботою Arduino IDE.
- **Програмне забезпечення з відкритим програмним кодом та розширюване програмне забезпечення –** Програмне забезпечення Arduino випускається у вигляді інструментів з відкритим початковим кодом, доступним для розширення програмістами з досвідом. Мова програмування може бути розширена за допомогою бібліотек C++ та

люди, що бажають зрозуміти технічні деталі можуть перейти від Arduino до мови програмування AVR-C, на якому він заснований. Точно так ви можете додати код AVR-C безпосередньо у ваші програми Arduino, якщо ви бажаєте.

- **Відкритий початковий код та розширюване обладнання –** Плани плат Arduino публікуються за ліцензією Creative Commons, тому досвідчені конструктори схем можуть створювати власну версію модуля, доповнюючи та покращуючи його. Навіть користувачі, що не мають великого досвіду можуть створювати макетну версію модуля для того, щоб зрозуміти як він працює та зекономити гроші. [4]

2.4 Вибір плати

Я вирішив розробити свій пристрій на базі плати Arduino Nano. Вона вирізняється серед інших плат своїм малим розміром, що значно допомагає у реальних проектах. Arduino Nano – це невелика сумісна та гнучка до макету плата мікроконтролера, розроблена італійським Arduino.cc на основі мікросхеми ATmega328p або ATmega168.

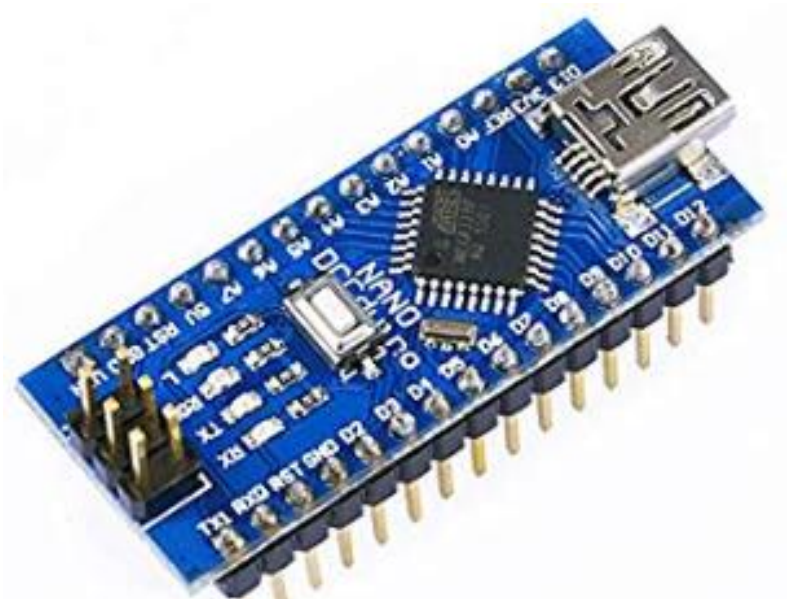


Рис.2.1. плата Arduino Nano

Arduino Nano Pinout містить 14 цифрових виводів, 8 аналогових виводів, 2 сброси та 6 виводів живлення.

- **Vin.** Це вхідна напруга живлення для плати при використанні зовнішнього джерела живлення від 7 до 12 В.

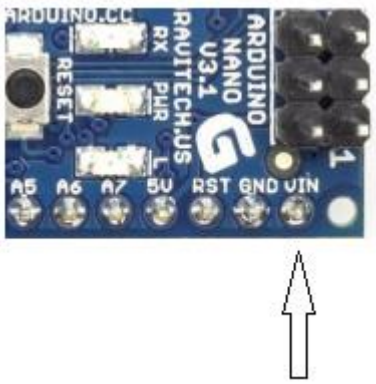


Рис.2.2 Vin

- **5V.** Це регульована напруга живлення плати, яка використовується для живлення контролера та інших компонентів, розташованих на платі.

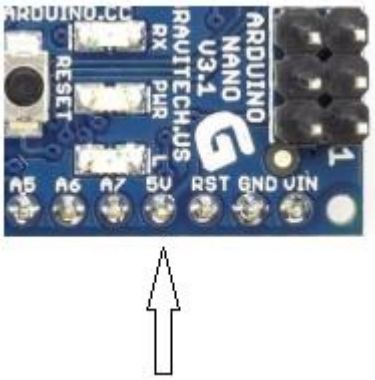


Рис.2.3 5V

- **3.3V.** Це мінімальна напруга, що генерується регулятором напруги на платі.

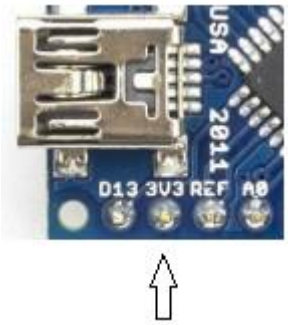


Рис.2.4 3.3V

- **GND.** Це виводи землі. На платі є декілька заземлених контактів які можуть бути відповідним образом з’єднані, коли необхідно більше одного заземленого контакта.

Рис.2.5 GND

- **Reset.** Пін сбросу додається для сбросу плати. Це дуже корисно, коли ввімкнена програма занадто важка та зависає. Низьке значення на виводі сбросу приведе до сбросу контролера.

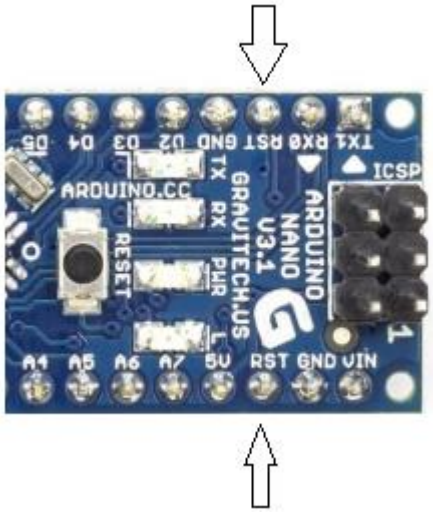


Рис.2.6 Reset

- **Analog Pins.** На платі є 8 аналогових виводів, що позначені як A0-A7. Ці контакти використовуються для вимірення аналогової напруги в діапазоні від 0 до 5 В.

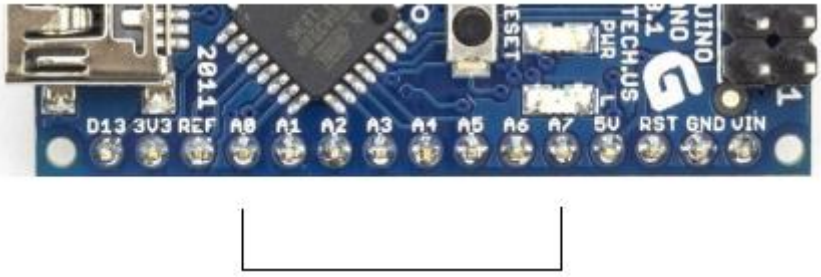


Рис.2.7 Analog Pins

- **Rx, Tx.** Ці контакти використовуються на платі для послідовного зв’язку, де Tx представляє передачу даних, в той час як Rx у свою чергу представляє приймач даних.

Рис.2.8 RXD, TXD

- **13.** Цей пін використовується для увімкнення вбудованого світлодіоду.
- **AREF.** Цей пін використовується в якості опорної напруги для вхідної напруги.
- **PWM.** Шість контактів 3,5,6,9,10,11 можуть використовуватися для забезпечення 8-канального виходу ШИМ(широтно-импульсної модуляції). Це метод, що використовується для отримання аналогових результатів з цифровим джерелом.
- **SPI.** Чотири контакти 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) використовуються для SPI (послідовний периферійний інтерфейс). SPI є інтерфейсною шиною та в основному використовується для передачі даних між мікроконтролерами та іншими периферійними пристроями, такими як датчики, регістри та SD-карти.
- **External Interrupts.** Контакти 2 та 3 використовуються як зовнішні переривання, які використовуються в разі надзвичайної ситуації, коли нам потрібно зупинити основну програму та викликати важливі інструкції у цей момент. Основна програма поновлюється після виклику та виконання інструкції переривання.
- **I2C.** Зв'язок I2C розроблений з використанням виводів A4 та A5, де A4 представляє собою лінію послідовних даних (SDA), яка переносить дані, а A5 представляє собою лінію послідовних тактових сигналів (SCL), яка в свою чергу є тактовим сигналом, що генерується ведучим приладом, що використовується для синхронізації даних між пристроями на шині I2C. [5]

2.5. Відмінність від інших плат

Компанія Arduino пропонує вам безліч плат на вибір для ваших проектів. Всі вони однакові за принципом своєї роботи тож ви можете замінити одну плату на іншу, у вашому проекті. Якщо звісно дозволить пам'ять

мікроконтролера та кількість виводів на платі. Частіше за все користувачі використовують Uno та Nano або ж плату Leonardo, якщо у цьому виникне необхідність.

Порівняння Nano та Uno

Головна відмінність між цими двома платами це їх розмір. Arduino Uno на декілька сантиметрів більша за Nano по ширині та довжині. Таким чином плата Uno займає більше місця в системі, що негативно впливає на розмір вашого девайсу. Якщо це має бути, щось портативне або носимий на руці пристрій така плата буде занадто збільшувати розмір пристрою. Uno в свою чергу є ідеальним для таких проектів, оскільки плата може поміститися у невеликий корпус та не буде псувати зовнішній вигляд пристрою, якщо це важливо у проекті. Спільним у цих двох плат є мікроконтролер ATmega328p який встановлений на обидві з них. Таким чином, вони можуть ділитися схожою програмою. Це стане у нагоді якщо ви, наприклад, вирішите відмовитися від Arduino Uno на користь Nano. Вам не доведеться переписувати свій код, та підлаштовуватися під іншу систему. На платах Arduino Uno використовується звичайний роз’єм USB (TYPE-B). В той час як Arduino Nano має на своїй платі роз'єм міні-USB. Але обидва ці кабелі є широко розповсюджені, тож скоріш за все вам не доведеться спеціально докуповувати один з них, якщо виникне така необхідність.

Порівняння Nano та Leonardo

Arduino Leonardo відрізняється від попередніх плат тим, що мікроконтролер ATmega32u4 на якому він працює, має вбудовану підтримку USB, що прибирає необхідність в додатковому чіпі процесора, на який ATmega328 опирається для передачі даних USB. Це дозволяє Leonardo відображатися на під’єднаному комп’ютері у вигляді миші чи клавіатури, на

додачу до віртуального COM-порту. Якщо вам знадобляться ці функції, а також весь функціонал плати Arduino Uno, те що вам потрібно це Leonardo.

Можна зробити висновок, що для проекту з моїм функціоналом, ідеально підійде плата Aruino Nano, яка володіє малим розміром та в той же час усіма необхідними можливостями. [6]

2.6. Вибір та опис складових компонентів

У моїй системі використовується декілька датчиків, та модуль WI-FI. Датчик температури це цифровий вимірювач температури DS18B20. Для вимірювання пульсу людини аналоговий датчик Pulse sensor. У якості WI-FI модуля була обрана платформа NodeMCU на основі ESP8266. Модуль вміє отримувати та передавати інформацію через мережу Інтернет. Тепер оглянемо детальніше усі складові.

DS18B20.Цифровий термометр, що забезпечує вимірювання температури в градусах Цельсія із здатністю 9-12 біт, та має тривожну функцію, яка буде сигналізувати якщо температура вийде за верхню або нижню межу, які встановлює сам користувач. Діапазон вимірюваних датчиком температур від -55° C до 125° C, з точністю ± 0.5 ° C в інтервалі від -10° C до +85° C. Окрім цього він може отримувати живлення прямо з лінії даних, що прибирає необхідінсть у зовнішньому джерелі живлення. Також може працювати у режимі «паразитного живлення». Це коли живлення приладу відбувається напругою якихось сигналів. Напруга яку бере прилад повинна бути досить малою аби не порушувати роботу сигнальної лінії

Кожний DS18B20 має свій унікальний 64-розрядний номер, який використовується для того, щоб дозволити багатьом датчикам працювати на одній однопровідній шині. Це дозволяє одному мікропроцесору керувати багатьма DS18B20, що розташовані на великій площі. Цим можна користуватися коли необхідно вимірювати температуру у будівлі, механізмів

машини або ж у палаті лікарні, якщо декілька пацієнтів під’єднати до одного приладу.

На рисунку 2.9 видно що датчик має 3 виводи.

- **GND** – земля.

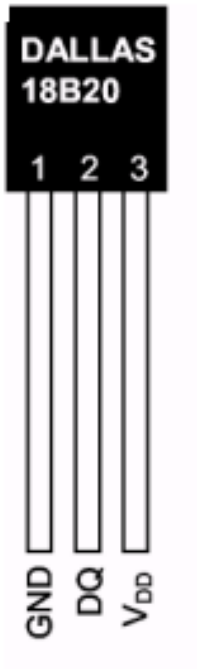


Рис.2.9 призначення виводів DS18B20

- **DQ** – вивід сигналу даних. Також саме через цей вивід відбувається живлення у режимі «паразитного живлення».
- **V_{DD}** – вивід зовнішнього живлення. У разі якщо використовується «паразитне живлення» повинен бути підключений до «землі».

Також необхідно пам’ятати, що не дивлячись на тип підключення, інформаційний вивід DQ слід під’єднувати до виводу живлення через резистор 4.7кОм. В разі якщо використовується лише один датчик, можна обрати резистор на 10кОм. [7]

Pulse sensor. Це імпульсний аналоговий датчик, що дозволяє зафіксовувати пульс людини. Цей датчик працює за рахунок встановлених на ньому світлодіоду та фотоприймачу. Вони розташовані таким чином, що на фотоприймач потрапляє промінь світла який відбивається від перешкоди

якою може бути палець, вухо, або зап’ястя. Схему роботи можна побачити на рисунку 2.

Судини, коли наповнюються кров’ю починають поступово змінювати свою оптичну щільність, що одразу впливає на зміну кількості відбитого світла. Отже, якщо рівень світлового потоку який випромінюється світлодіодом буде постійний, інтенсивність світла яка може реєструватися

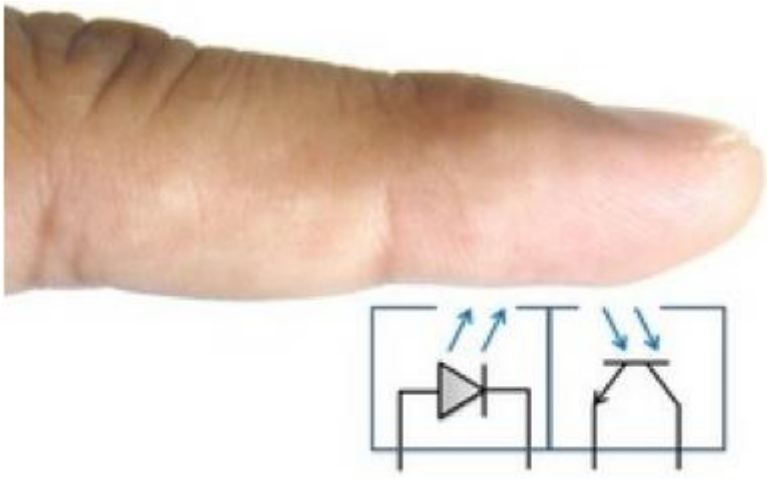


Рис.2.10 Принцип роботи датчику

фотоприймачем буде залежати від наповненості сосудів кров’ю.[8]

Електрична схема датчика пульсу розроблена спеціально, щоб реєструвати тільки динамічну зміну інтенсивності приймаємого світлового потоку. Якщо світловий потік є незмінним (незалежно від його інтенсивності), напруга на виході датчика коливається в районі половини напруги живлення. У разі якщо відбувається зміна світлового потоку, напруга на виході датчика відхиляється від середнього значення, в бік зменшення або збільшення, пропорційно зміні світлового потоку. Таким чином датчик не потребує налаштування під кожну людину.

Датчик пульсу має розмір приблизно 16 міліметрів в діаметрі та близько 3 міліметрів завтовшки. Напруга живлення на датчику 5 В. А струм споживання 4 мА.

Рис.2.11 Виводи датчику Pulse Sensor

На рис.2.11 ви можете побачити 3 виводи на задній поверхні датчику: s, +, -.

- **S** : Це сигнальний вивід, що під'єднується до будь-якого аналогового контакту мікроконтролеру.
- **+** : Це живлення, від 3 до 5 В.
- **-** : Це земля, під'єднується до виводу GND мікроконтролера. [9]

NodeMCU V3 ESP8266. Модуль який базується в основному на ESP8266 – це недорогий Wi-Fi мікрочіп, що включає як повний стек TCP/IP, так і можливість мікроконтролера. Він представлений виробником Espressif Systems – виробником, що базується в Шанхаї, Китай. Сам чіп спроектований для пристроїв зі світу Інтернет Речей, а дана плата дозволяє спростити розробку, так як на ній вже реалізоване підключення через USB, регулятор живлення, а всі виводи чіпа розведені не гребінки зі стандартним кроком 2,54 мм, що дозволяє вставляти його в макетну плату та створювати прототип навіть не вмикаючи паяльник. Окрім того плата одразу поставляється з прошивкою NodeMCU, що дозволяє програмувати її за допомогою мови Lua або за допомогою Arduino IDE.

Плата має такі характеристики:

- Wi-Fi стандарту 802.11 b / g / n
- Підтримка STA / AP / STA + AP режимів
- Струм на виводі 15мА
- Живлення від 4,5 до 10В максимум
- Швидкість передачі даних 110-460800 бод за секунду
- Діапазон робочих температур від -40 до +125 градусів по Цельсію
- Маса плати 18 г [10]

Опис та призначення виводів

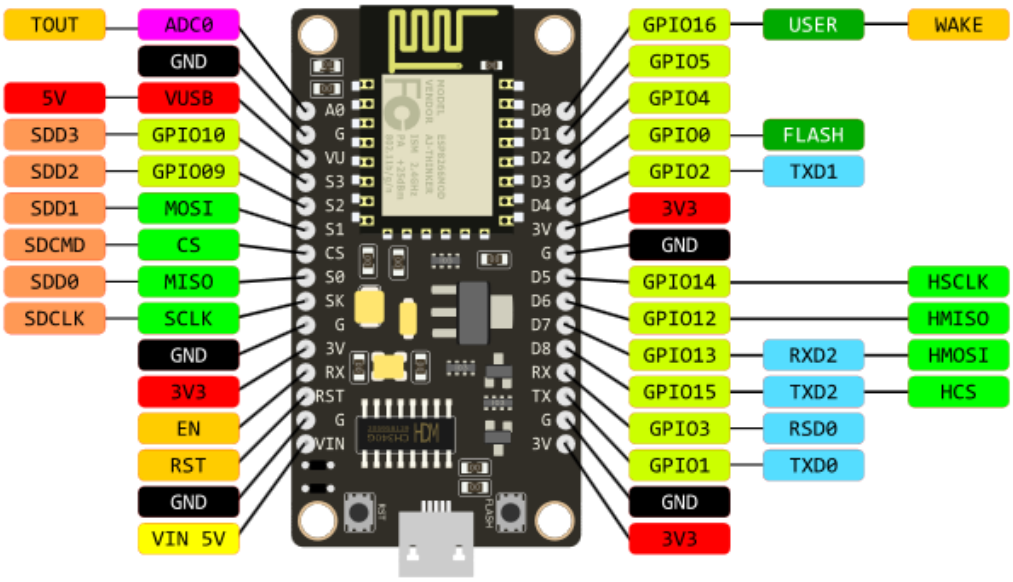


Рис.2.12 Призначення виводів NodeMCU

GND. Як видно на рисунку 2.12 на платі є 5 виводів GND, тобто «земля».

Виводи живлення. Vin – вивід для підключення зовнішнього джерела живлення 5В. Стабілізатор AMS1117-3.3 дозволяє подавати живлення на Vin в широкому діапазоні від 5 до 10 В. 3.3V – контакт на який подається вихідна напруга схемного стабілізатора. Може бути використаний для живлення підключаємих до плати датчиків. Сумарне максимальне навантаження усіх

виводів 3.3V не потрібне перевищувати 300 мА. V USB – вивід на який заведена напруга 5V з USB-роз'єму.

Виводи GPIO. General Purpose Interput Output – контакти вводу/виводу загального призначення. Можуть бути сконфігуровані як входи та виходи та програмно призначені на різні функції.

Виводи управління. Reset – вивід використовується для сбросу мікроконтролера ESP8266. EN – при подачі на контакт сигналу високого рівня, мікроконтролер ESP8266 переходить в робочий режим, при сигналі низького рівня – в режим енергозбереження. WAKE – контакт використовується для пробудження чіпа ESP8266 з режиму глибокого сну.

АЦП (ADC). ADC0/TOUT – вивід вбудованого 10-розрядного аналого-цифрового перетворювача. Перетворені значення лежать в інтервалі 0-1023. Плата розробки NodeMCU V3 поставляється з внутрішнім дільником напруги, тому вхідний діапазон становить від 0 до 3.3 В. Діапазон вхідної напруги для АЦП в кристалі ESP8266 від 0 до 1 В.

UART. Асинхронний послідовний інтерфейс встановлює зв'язок з іншими пристроями по шині UART. У контролера ESP8266 два модулі UART. Максимальна швидкість передачі даних, що заявляється виробником 4,5 Mbps.

SPI. Послідовний периферійний інтерфейс NodeMCU має два SPI (SPI та HSPI) у ведучому та підлеглому режимі.

SDIO. Інтерфейс безпечних цифрових входів/виходів, призначений для комутації з зовнішньою флеш пам'яттю стандарту SD по послідовній шині.

Reserved. Зарезервовані виводи.

FLASH. Кнопка Flash на NodeMCU підключає до землі GPIO0. Її можна використовувати як звичайну кнопку. Якщо програмно підтягнути вивід GPIO0 за допомогою внутрішнього підтягуючого резистору до високого рівня, то поява низького рівня на цьому виводі буде сповіщати про те, що кнопка натиснута.

Інтерфейс I2C. Послідовна асиметрична шина. I2C використовується для підключення датчиків та периферійних пристроїв. NodeMCU ESP8266 не має апаратних виводів I2C , але інтерфейс можливо реалізувати програмно. Підтримуються як I2C Master, так й I2C Slave. Зазвичай в якості контактів I2C використовуються наступні виводи: GPIO5 – SCL, GPIO4 – SDA.

PWM (pulse-width modulation). Широтно-імпульсна модуляція (ШІМ) керує потужністю за допомогою метода пульсуючого ввімкнення та вимкнення виводу. NodeMCU підтримує програмний ШІМ на виводах, що позначені на рисунку нерівною лінією. [11]

2.7. Вибір середовища розробки

Arduino популярна платформа, а отже існує велика кількість середовищ розробки які дозволяють запрограмовувати мікроконтролери. Кожен має свої переваги і недоліки, та різний функціонал й спосіб роботи. Я хочу оглянути 4 найпопулярніші середовища розробки та обрати найкращий для мне варіант.

Середовище Programino

Розглянемо середовище Programino. Це платне середовище розробки але його можна випробувати протягом 14-ти днів безкоштовно. Programino, як й інші середовища розробки потребує щоб у вас була встановлена Arduino IDEю При першому запуску програми необхідно в налаштуваннях вказати шлях до виконуваного файлу arduino.exe. Мова яка використовується в цьому середовищі така ж як і в оригінальній Arduino Ide, це мова C. Отже, по суті, якщо ви вже пишете скетчі в Arduino IDE, то вам не доведеться вивчати нову мову програмування, що є великим плюсом даного середовища. Однак крім цього, дана IDE пропонує такий зручний спосіб швидкої розробки як автозаповнення коду. Тобто вам не доведеться постійно відкривати довідник по командам та методам Arduino. Ви починаєте набирати код, а середовище

розробки запропонує вам обрати з декількох варіантів той, який вам потрібен. Наприклад ви набираєте “digi” і IDE пропонує вам варіанти:“digitalRead”, ”digitalWrite” та інші можливі.

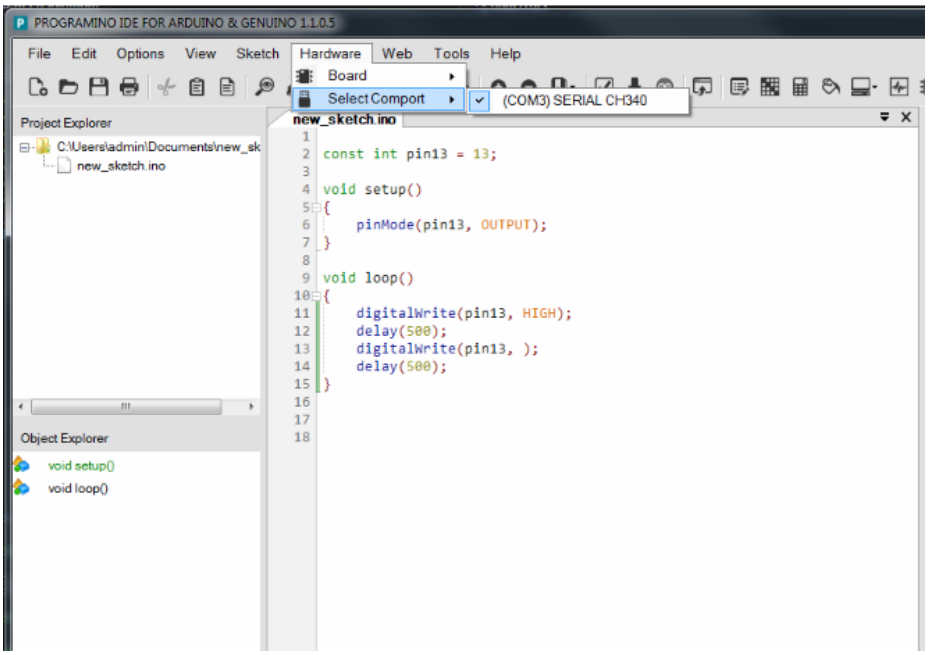


Рис.2.13 Середовище розробки Programino

Середовище B4R (Basic for Arduino)

Ще одна цікава альтернатива Arduino IDE – Basic for Arduino. Ця середа розробки унікальна тим, що використовує мову Basic, а не C. Вона також підтримує функцію автодоповнення коду. Окрім того, вона повністю безкоштовна. При першому запуску середовище B4R також потребує вказати шлях до директорії з Arduino IDE та при необхідності додатковими нестандартними бібліотеками та загальними модулями. Ви не можете одразу почати програмувати в цьому середовищі розробки так як вона використовує інший, більш об’єктно-орієнтовану мову програмування, ніж класична Arduino IDE, з іншим синтаксисом. Тим не менш, зручність цього средовища та наявність гарного довідника від розробників повністю перемагає ці недоліки. [12]

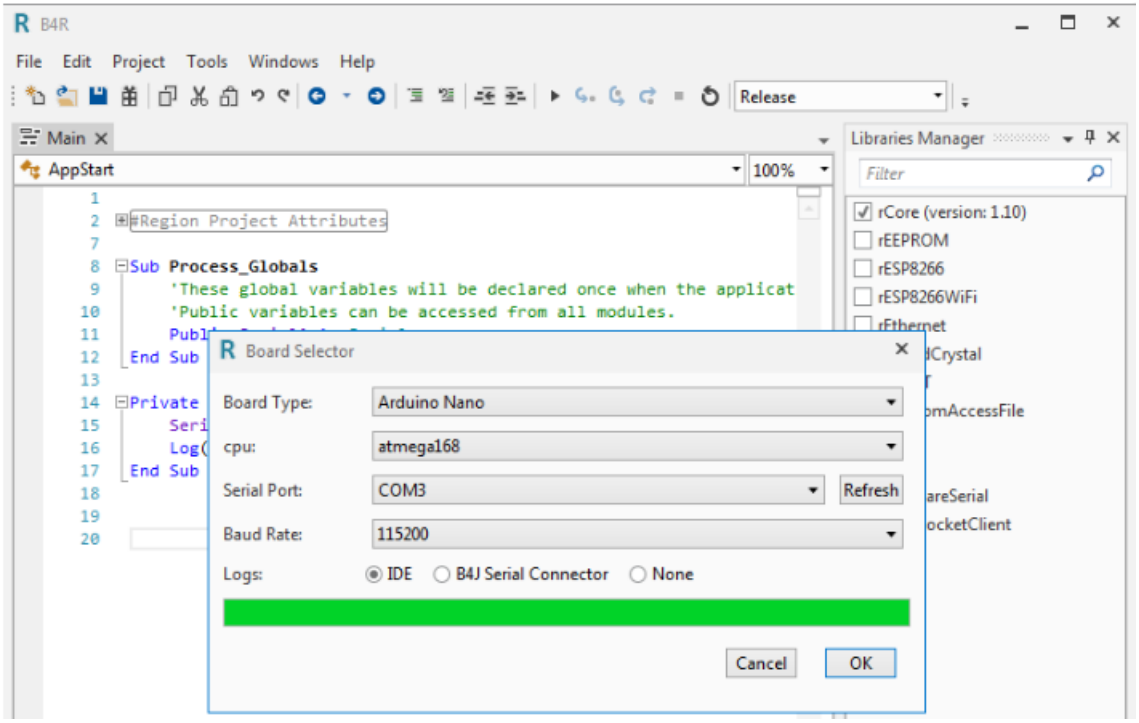


Рис.2.14 Середовище розробки B4R

Середовище Codeblocks for Arduino

Існують й інші середовища розробки для Arduino окрім названих вище. Наприклад, CodeBlocks. Основна її відмінність від описаних IDE – це можливість писати код для мікроконтролерів та деяких інших платформ, а не тільки для Arduino. В дистрибутів інтегровані останні файли ядра Arduino, стандартні бібліотеки Arduino, набір інструментів AVR, Arduino Builder, послідовний термінал і що найцікавіше симулятор Arduino рівня API, але він знаходиться в стадії розробки. Можна відмітити такі функції цього середовища: спеціальний проект-майстер для розробки Arduino, скомпільовані файли ядра кешовані для більш швидкої компіляції (у порівнянні з оригінальною Arduino IDE), вбудований попередньо сконфігурований AVR-компілятор, завантаження HEX на дошки Arduino (підтримується платою Leonardo) запускаючи вбудовану мішень, а також в якості цілей сборки підтримуються всі популярні плати Arduino IDE. [13]

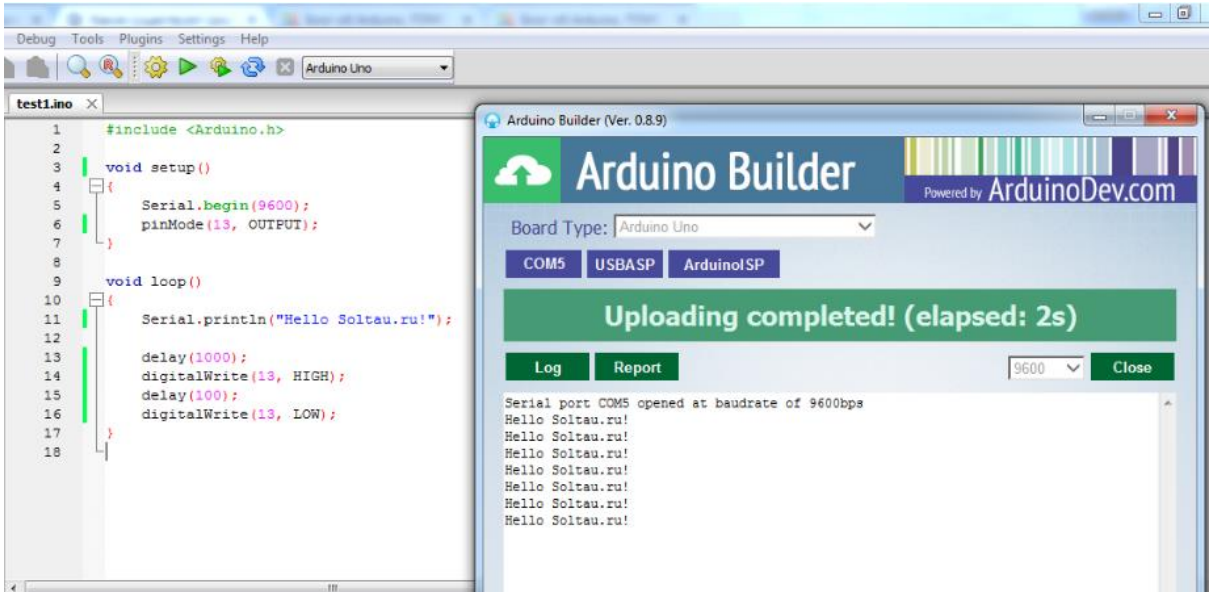


Рис.2.15 Середовище розробки Codeblocks

Середовище Arduino IDE

Звісно ж не можна забувати про найпопулярніше середовище розробки, яке розробила сама компанія Arduino. В ньому є весь необхідний мінімум для розробки програм: написання коду, перевірка коду, компіляція, завантаження скетчу в Arduino, монітор послідовного порту. Всі хто працював в серйозних середовищах розробки по типу JetBrains IDEA, Microsoft Visual Studio або Quartus, напевно відмітить, що середа Arduino IDE досить аскетична, нічого зайвого і ніяких зручностей вона не пропонує.

Скетчі це програми які ми створюємо в середовищі Arduino IDE. Вину пишуться за допомогою текстового редактора та потім можуть зберігатися в файлах з .ino розширенням. Текстовий редактор який присутній у середовищі володіє усім необхідним функціоналом, а саме: заміна тексту, копіювання, вставка та пошук. Також в вікні програми присутня область повідомлень яка може давати користувачу зворотній зв’язок, та надсилати йому повідомлення про процеси, та помилки які виникають в системі. [14]

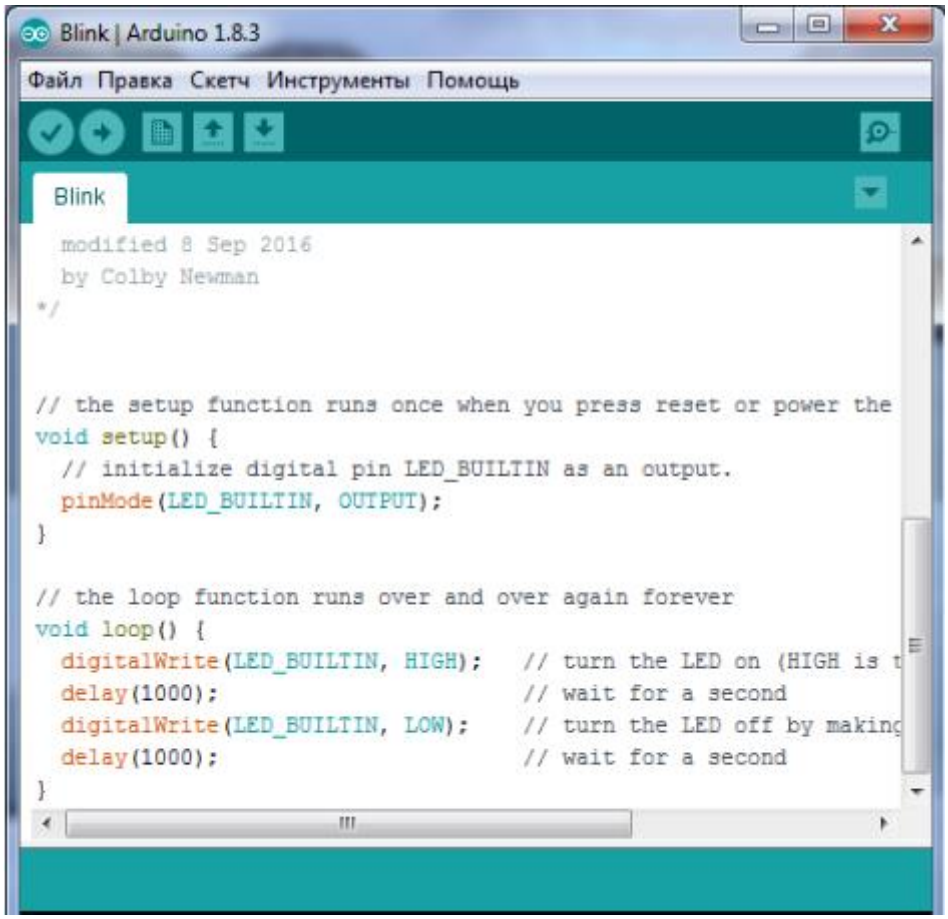


Рис.2.16 Середовище розробки Arduino IDE

Отже, в якості середовища розробки я буду використовувати саме Arduino IDE. Воно володіє всіма необхідними функціями, та не має зайвих які могли б заважати зосередженій роботі з програмою. Також важливою перевагою є те що це середовище розробляла та сама ж компанія, отже воно офіційне, і не потребує додаткових обхідних налаштувань, як цього вимагають інші IDE.

ВИСНОВКИ ДО РОЗДІЛУ 2

В цьому розділі я провів дослідження та пошук складових компонентів необхідних для створення пристрою стеження за станом пацієнта. Було детально оглянуто їхні характеристики, ти визначено слабкі та сильні сторони кожного з компонентів пристрою.

Серед багатьох датчиків які можуть фіксувати пульс та температуру тіла було обрано саме ці через їх точність та не дуже високу ціну. Звичайно вони не володіють точністю професійних медичних апаратів, але для демонстрації роботи проекту вони цілком підходять. Якщо необхідна більша точність то можна легко придбати більш якісні датчики, та замінити їх у пристрої. В цьому і полягає великий плюс Arduino систем.

Стає зрозуміло чому серед багатьох плат Arduino які представлені на ринку було обрано саме Arduino Nano. Вона досить мала за розміром, одна з найменших плат які розробляє ця компанія. А якщо порівнювати її розмір з функціоналом то вона є найкращою з представлених плат.

Також розглядаються і порівнюються середовища розробки для плат Arduino. Після ознайомлення з функціоналом чотирьох найпопулярніших вибір пав на класичне Arduino IDE, перевага якого над іншими середовищами детально описана у розділі.

Отже в цьому розділі було розглянуто плати, датчики та середовища розробки. Обрано ті що підходять найбільше і написано чому саме вони.

РОЗДІЛ 3
МОДЕЛЮВАННЯ І КОНСТРУЮВАННЯ ПРИСТРОЮ

3.1. Моделювання плати

В першу чергу необхідно розробити схему підключення усіх датчиків. Датчики температури та пульсу повинні бути підключені до плати Arduino Nano, яка буде приймати дані. Вона в свою чергу до модуля ESP для відправки даних.

Pulse sensor під’єднується до виводів «землі» GND, живлення датчика 5V та аналогового виводу A0.

Датчик DS18B20 під’єднується до контактів живлення 3V, землі (GND) та цифровий контакт D2.

Wi-Fi модуль ESP8266 під’єднується до плати Arduino Nano через цифрові контакти D5 та D6. Через ці виводи буде передаватися інформація з однієї плати на іншу.

На схемі нижче зображений пристрій, який складається з таких елементів:

1. Pulse sensor
2. Датчик температури DS18B20
3. Резистор на 4.7кОм
4. Arduino Nano
5. Модуль Wi-Fi NodeMCU ESP8266

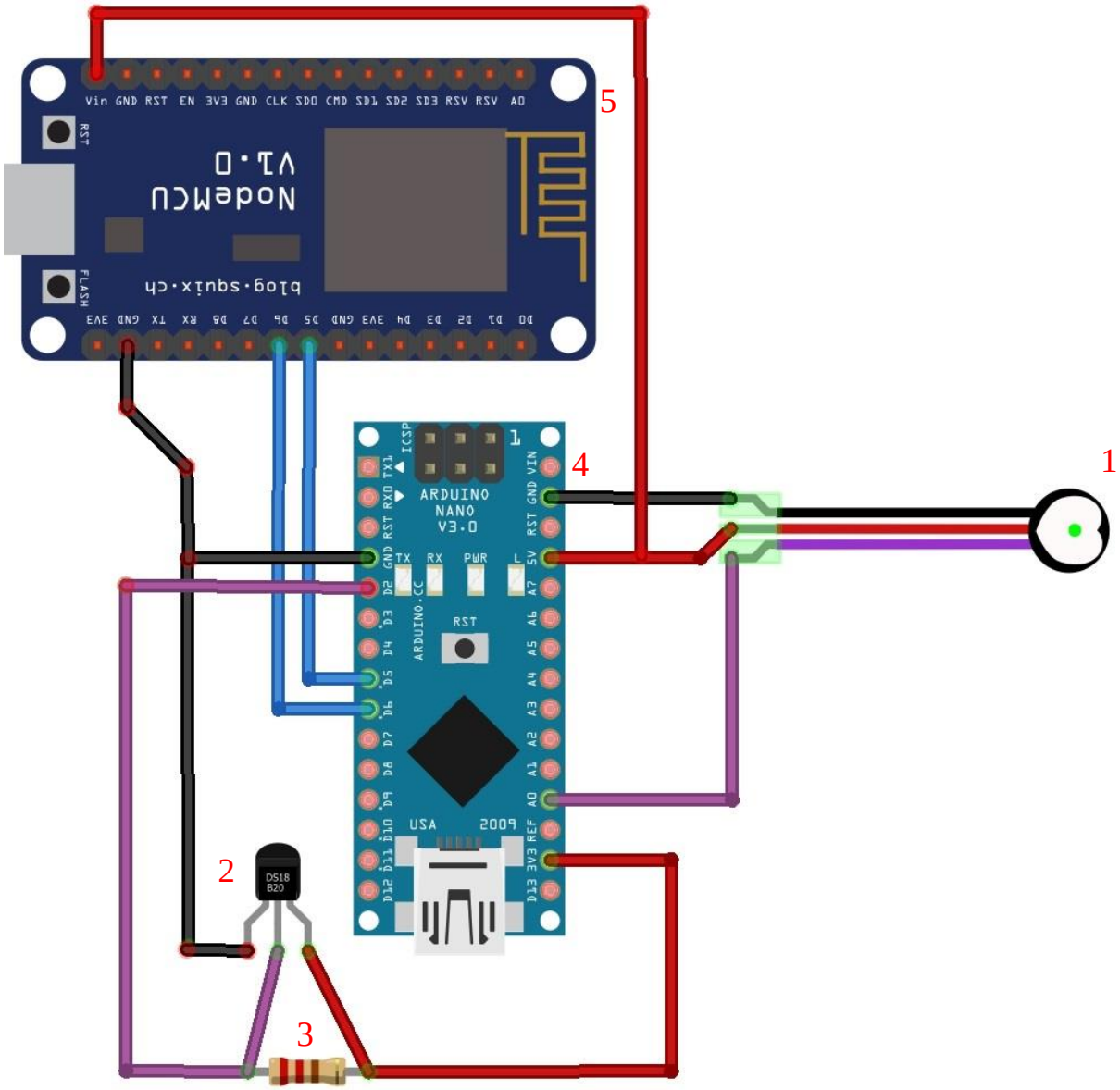


Рис. 3.1. Схема пристрою

3.2. Програмування девайсу

Після того як ми спроектували та під’єднали усі елементи пристрою, настає черга розробити програмний код. У Arduino він завантажується у пам’ять мікроконтролера. Оскільки в нас їх два то код який повинен працювати за датчиками ми завантажимо на Arduino Nano, а код який буде відправляти всі дані на Google Диск завантажимо на мікроконтролер з модулем Wi-Fi ESP8266.

Програмування Arduino Nano

Спочатку потрібно розібратися які функції повинен виконувати код який буде завантажений на Arduino Nano. По-перше, він повинен у режимі реального часу отримувати дані з датчиків температури та пульсу. Та потім кожні 8 секунд відправляти дані на Wi-Fi модуль.

```
#define USE_ARDUINO_INTERRUPTS true
#define ONE_WIRE_BUS 2
```

Рис. 3.2 Запис константи #define

Директива #define, що зображена на рис.3.2, використовується для задання імені константам перед компіляцією програми. Константи, що задані цією директивою не займають програмної пам'яті, оскільки компілятор заміняє всі звернення до них, їх значенням на етапі компіляції. Отже вони служать для зручності програміста та більшої читабельності тексту. Нам важливо задати константі USE_ARDUINO_INTERRUPTS значення true для налаштування низьких рівнів переривань для найбільш точної математики пульсу та оскільки цього вимагає бібліотека PulseSensorPlayground.h яку я використовую для роботи з датчиком пульсу. А задання ONE_WIRE_BUS значення 2 вказує, що контакт для даних ми під'єднали до цифрового піна 2.

```
#include <SoftwareSerial.h>
#include <PulseSensorPlayground.h>
#include <OneWire.h>
#include <DallasTemperature.h>
```

Рис.3.3 Підключення сторонніх

#include використовується для імпортування сторонніх бібліотек у наш скетч. Вона дає доступ до більшого числа стандартних бібліотек та бібліотек написаних спеціально для Arduino. [15]

Бібліотека SoftwareSerial.h дозволяє реалізувати послідовний інтерфейс на будь-яких цифрових виводах Arduino за допомогою програмних засобів. Вона дозволяє програмно створювати декілька послідовних портів, що працюють на швидкості до 115200 бод. Бод – це певна кількість імпульсів яка

може передаватися за 1 секунду. Саме за допомогою неї дані будуть передаватися з плати Nano на модуль ESP8266. За допомогою бібліотеки PulseSensorPlayground.h я буду контролювати датчик пульсу, та зчитувати з нього інформацію. Бібліотека OneWire.h необхідна для керування датчиками які виробляються компанією Dallas, саме вона створила датчик температури який я використовую. DallasTemperature.h використовується сумісно з OneWire.h. [16]

```
SoftwareSerial s(5,6);

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
PulseSensorPlayground pulseSensor;

const int PulseWire = 0;
float Threshold = 550;
```

Рис.3.4 Оголошення змінних

В першому рядку коду ми оголошуємо, що передавати дані між платами будемо через 5 та 6 цифровий піни. Потім ми створюємо екземпляр класу OneWire для того, щоб з його допомогою обмінюватися сигналами з однопроводними пристроями, у нашому випадку це температурний датчик. Далі передаємо об’єкт OneWire об’єкту sensors. В наступному рядку створюємо екземпляр об’єкту PulseSensorPlayground під назвою pulse sensor. На сам кінець, створюємо дві змінні одна з яких вказує на те, що датчик пульсу буде зчитуватися з нульового аналогового піну. Інша змінна використовується для визначення того який сигнал буде рахуватися як удар серця, а який ігноруватися.

```
void setup() {
  delay(10000);
  s.begin(9600);
   Serial.begin(9600);


  pulseSensor.analogInput(PulseWire);
  pulseSensor.setThreshold(Threshold);

if (pulseSensor.begin()) {
    Serial.println("Start pulse");
   sensors.begin();
}

}
```

Рис.3.5 Функція setup

Функція setup необхідна для ініціалізації системи. У цій функції знаходяться команди які мікроконтролер виконає один раз в момент завантаження при запуску системи. Delay(1000) «каже» мікроконтролеру чекати 10 секунд перш ніж виконувати наступні команди. Це необхідно для того, щоб датчики встигли налаштуватися та одразу ж почали показувати вірний результат. S.begin(9600), ця інструкція ініціалізує роботу з послідовним портом, який буде передавати дані на іншу плату, на швидкості у 9600 імпульсів в секунду. Далі використовуючи pulseSensor, який є екземпляром об’єкту PulseSensorPlayground я вказую мікроконтролеру з якого аналогового піна будуть зчитуватися дані з датчику пульсу. У моєму випадку це пін A0. Потім вказую які дані пульсу будуть ігноруватися. Через умовний оператор if я вказую програмі, що бібліотека sensors повинна запускатися лише після того як датчик пульсу настроїться, та почне передавати актуальні дані.

```
void loop() {
  delay(8000);
  int myBPM = pulseSensor.getBeatsPerMinute();
  if (pulseSensor.sawStartOfBeat()) {
 Serial.print("Pulse: ");
 Serial.println(myBPM/2.7);
}
```

Рис.3.6 Функція loop

У функцію loop ми повинні помістити ті команди які будуть виконуватися весь час, доки плата Arduino увімкнена. Почавши з першої команди мікроконтролер по черзі проходить всі, поки не дійде до самого кінця, виконавши останню команду він «перестрибує» на початок та повторює цю послідовність. І так нескінченну кількість разів, доти доки на плату буде подаватись напруга. Тобто void loop це головна функція входу в нашу програму. Arduino повторює виклик цієї функції мільйони раз в секунду, таким чином ви можете бути впевнені, що програма буде виконуватися.[17]

Delay(8000) необхідний для того щоб команда виконувалася один раз за 8 секунд, це покращує точність передаваних даних, та робить їх обробку зручнішою. В наступному рядку коду я присвоюю змінній myBPM значення типу int кількості ударів серця за хвилину, використовуючи функцію getBeatsPerMinute().

```
  sensors.requestTemperatures();
  float t = sensors.getTempCByIndex(0);
Serial.print("TEMPERATURE = ");
Serial.print(t);
Serial.print("*C");
Serial.println();
```

Рис.3.7 Отримання температури

Потім через умовний оператор if та функцію sawStartOfBeat я перевіряю чи серцебиття сталося. Якщо програма отримує позитивний результат то на

екран виводиться значення пульсу, а контролер продовжує виконання завдання.

На рис.3.7 запит sensors.requestTemperatures() звертається до всіх пристроїв на шині для видачі даних температури. Після того як ми отримали температуру, то можемо надрукувати її або ж переслати на Wi-Fi модуль. getTempCByIndex(0) це функція яка повертає температура за індексом, отже ми отримаємо температуру лише від першого датчика. Це корисно, якщо до шини приєднано декілька температурних датчиків. Значення температури записується у змінну t з типом даних float.

На рис.3.8 зображений програмний код який пересилає дані на ESP8266.

```
if(s.available()>0)
{

 s.write(myBPM/2.7);

}

if(s.available()>0)
{

 s.write(t);

}
```

Рис.3.8 Пересилання даних

Точніше він записує значення пульсу та температури в змінну, яку потім зчитує NodeMCU. Умовний оператор if з командою s.available() необхідний для того щоб перевірити чи є вхідні дані в серії. Отже спочатку виконується перевірка на наявність вхідних даних. Якщо відповідь позитивна то дані записуються та передаються на іншу плату. Спочатку відправляється результат моніторингу пульсу, а одразу за ним дані температури тіла. [18]

Підсумовуючи функції які виконує програмний код, що завантажується на Arduino Nano. Мікроконтролер підключається до датчиків та перевіряє чи

почала надходити від них інформація. Після отримання необхідних даних він надсилає ці дані на NodeMCU.

Програмування NodeMCU

Спочатку потрібно вирішити які задачі стоять перед цим мікроконтролером. Перша функція це отримання результатів діагностики з іншої плати. Вони будуть отримуватися кожні 8 секунд та записуватися у свої змінні. Також плата повинна відправляти дані на Google Диск. Для цього необхідно приєднуватися до мережі Wi-Fi. Оскільки необхідно, щоб в процесі аналізу результатів лікар бачив у який час робилися виміри, необхідно разом з результатами моніторингу відправляти й час заміру. З комп'ютера отримати час не вдасться, тому що, пристрій повинен працювати автономно. Рішенням є отримувати час з серверу світового часу. Кожен раз коли виріб буде відправляти інформацію на Google Диск він також разом з цим буде приєднуватися до сервера, отримувати час, переводити у зручний формат та завантажувати на Google Диск.

```
#include <SoftwareSerial.h>
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
#include "HTTPSRedirect.h"
#include "DebugMacros.h"

SoftwareSerial s(D6,D5);
```

Рис.3.9 Підключення необхідних бібліотек

Окрім програмування NodeMCU необхідно буде налаштувати та запрограмувати Google Диск для того щоб він міг отримувати інформацію з Arduino, але це згодом.

Про бібліотеку SofwareSerial.h я розказував коли описував програмування Arduino Nano, тут вона виконує ті самі функції. ESP8266Wi-Fi.h це та бібліотека яку ви використовуєте для підключення свого модулю до мережі Wi-Fi, щоб почати відправку або отримання даних. Наступна

бібліотека буде використовуватися для програмування UDP-функціоналу. UDP або протокол датаграм користувача це один з ключових мережевих протоколів для Інтернету. Він дозволяє комп'ютерним додаткам надсилати повідомлення іншим IP-хостам без необхідності попередньої установки каналів зв'язку. Бібліотека HTTPSRedirect.h заснована на бібліотеці Wifi. Вона надає користувачу можливості для легкого спілкування з серверами через http та https. Для зв'язку по протоколу HTTPS однією з вимог є сертифікати підключення або якщо можна так сказати «відбитки пальців» сайтів. Бібліотека DebugMacros.h допомагає зберігати та обробляти запити в цьому рядку. Та на сам кінець ми оголошуємо, що сигнали між платами будуть передаватися через цифрові піни D6 та D5. [19]

```
unsigned int puls;
unsigned int temp;
String sheetPulse = "";
String sheetTemp = "";
const char* ssid = "TP-Link_BA54";
const char* password = "slastion4";          //Put WiFi password within the quotes
const char* host = "script.google.com";
const char *GScriptId = "AKfycbzqZP-nVgwlDcSAQYdtFsza__IUCySWWIw4o0KjDLFfhIAdxBSO"
const int httpsPort = 443; //the https port is same
```

Рис.3.10 Задання змінних

Спочатку ми вказуємо тип змінних які будуть зберігати значення пульсу та температури unsigned int. Цей тип схожий на int тим що так само зберігає двухбайтові значення. Різниця в тому, що замість від'ємних чисел він може зберігати лише додатні значення в діапазоні від 0 до 65535.

Далі необхідно створити змінні з типом String, в які потім запишуться дані моніторингу. Ці змінні будуть об'єднані в одну яка потім буде передаватися на Google Диск. Потім задаються змінні які зберігають інформацію необхідну для підключення до Wi-Fi та Google Диску. Вони будуть типом char, який займає в пам'яті 1 байт та зберігає символьні значення. Однак у пам'яті символи зберігаються як числа. Відповідне кодування ви можете знайти в таблиці ASCII-код символів. Змінні ssid та

password зберігають назву та пароль від точки доступу до Інтернету. Їх можна змінювати якщо пристрій буде використовуватися у іншій мережі. У змінній host зберігається назва сайту, доступ до якого має отримувати мікроконтролер, для завантаження даних.

Змінна GScriptId зберігає в собі ідентифікатор скрипту, вбудований в URL веб-додатку. Вона використовується кодом Arduino для доступу до листа Google. Цей код можна отримати при налаштуванні Google Диску. За замовчуванням для https використовується порт 443, саме таке значення надається змінній httpsPort.

```
const char* ntpServerName = "0.ua.pool.ntp.org";
const int NTP_PACKET_SIZE = 48; // NTP time stamp
byte packetBuffer[ NTP_PACKET_SIZE]; //buffer to
```

Рис.3.11 Задання змінних серверного часу

У змінній ntpServerName, як показано на рисунку 3.11 буде зберігатися ім’я серверу на який мікроконтролер буде надсилати запит для отримання реального часу. Це український сервер тож час запиту буде мінімальний. NTP_PACKE_SIZE має значення типу int яке дорівнює 48. Це тому що відмітка часу у відповіді сервера знаходиться у перших 48 байтах повідомлення.

NTP (network time protocol) це мережевий протокол даних часу і це інтернет протокол який використовується для синхронізації годинника комп’ютера з певним точним часом. NTP – це стандартний протокол Інтернету, який був створений професором Девідом Мілсом в університеті в штаті Делавер. Я ж в свою чергу використовую NTP протокол не для синхронізації часу комп’ютера як він для цього розроблений, а для того щоб отримувати реальний час, який я міг заносити в електрону таблицю, поряд з даними вимірів. [20]

В кінці оголошуємо змінну буфер в якій будуть зберігатися вхідні та вихідні повідомлення. Буферні змінні необхідні лише для обміну даними, саме тому вони мають таку назву

На рис.3.12 зображений фрагмент коду де я спочатку використовую клас

```
client = new HTTPSRedirect(httpsPort);
client->setInsecure();
client->setPrintResponseBody(true);
client->setContentTypeHeader("application/json");
Serial.print("Connecting to ");
Serial.println(host);

bool flag = false;
for (int i = 0; i < 5; i++) {
  int retval = client->connect(host, httpsPort);
  if (retval == 1) {
    flag = true;
    break;
  }
  else
    Serial.println("Connection failed. Retrying...");
}
if (!flag) {
  Serial.print("Could not connect to server: ");
  Serial.println(host);
```

Рис.3.12 Створення TLS з’єднання

HTTPSRedirect для створення нового TLS з’єднання. TLS (transport layer security) – це протокол захисту транспортного рівня. Якщо точніше то це криптографічний протокол, що забезпечує захищену передачу даних між вузлами в мережі Інтернет. Він використовує асиметричне шифрування для автентифікації, симетричне шифрування для конфіденційності та коди

автентичності повідомлень для збереження цілісності повідомлень. Далі я намагаюсь під'єднатися до script.google.com, щоб мати можливість пересилати змінні. Через умовний оператор if я обмежую кількість спроб підключення до 5 раз, якщо не вийде то припинити виконання коду. [21]

```
WiFi.hostByName(ntpServerName, timeServerIP);

sendNTPpacket(timeServerIP);

delay(1000);
```

Рис.3.13 Відправка NTP пакета

На рис.3.13 в першій строчці коду програма під'єднується до серверу, ім'я якого було задано раніше, при ініціалізації змінних. Далі за допомогою команди sendNTPpacket відправляємо на сервер реального часу NTP пакет. Та чекаємо 1 секунду, щоб дізнатися чи з'явилася відповідь. NTP (network time protocol) протокол мережевого часу для синхронізації внутрішнього годинника комп'ютера з використанням мереж зі змінною латентністю. NTP, що заснований на алгоритмі Марзулло, використовує для своєї роботи протокол UDP та враховує час передачі. Також необхідно сказати, що система NTP дуже стійка до змін латентності серед передачі. В четвертій версії може досягати швидкості 10 мс (1/100 с) при роботі через Інтернет, та до 0.2 мс (1/5000 с) й краще всередині локальних мереж.

```
int cb = udp.parsePacket();
if (!cb) {
  Serial.println("no packet yet");
} else {
  Serial.print("packet received, length=");
  Serial.println(cb);

  udp.read(packetBuffer, NTP_PACKET_SIZE);
```

Рис.3.14 Зчитування NTP пакета

Udp.parsePacket() перевіряє наявність прийнятого UDP-пакету та повертає його розмір. Функція parsePacket() повинна визиватися перед зчитуванням буфера методом udp.read(). За допомогою parsePacket() ми ініціалізуємо змінну cb як цей пакет. Якщо вона порожня виводиться повідомлення, що отримати пакет не вдалося. Інакше, виводиться повідомлення про вдалу спробу та сама дані зчитані з цього пакету.

```
String hour = String(((epoch  % 86400L) / 3600)+3);
Serial.print(':');
String minute = "";
if (((epoch % 3600) / 60) < 10) {
  // In the first 10 minutes of each hour, we'll want
  Serial.print('0');
  minute = String('0')+ String((epoch % 3600) / 60);
} else {
  minute = String((epoch % 3600) / 60);
}
String second = "";
Serial.print((epoch  % 3600) / 60); // print the minu
Serial.print(':');
if ((epoch % 60) < 10) {
  // In the first 10 seconds of each minute, we'll wa
  second = String('0') + String(epoch % 60);
  Serial.print('0');
} else {
  second = String(epoch % 60);
}
```

Рис.3.15 Розрахунок часу

Потім програма зчитує дані в буфер за допомогою функції udp.read(). Якщо параметри в ній не задані, то вона повертає черговий прийнятий символ.

Змінна epoch на рис.3.15 зберігає в собі теперешній час, у форматі секунд, починаючи з 1970 року. Для того щоб спочатку дізнатися точний час в годинах я цю змінну ділю на 86400, це кількість секунд в дні. А потім на 3600, для того щоб дізнатися кількість годин не в секундах. Потім необхідно знайти час в хвилинах. Для цього змінну epoch ділимо на кількість годин в секундах. Дізнавшись це число потрібно поділити його на 60 секунд. Кінцеве число буде точним часом у хвилинах. За допомогою умовного оператору if я перевіряю, якщо число менше 10 то перед кількістю хвилин буде додано число 0, щоб час складався з 2 цифр. На сам кінець потрібно дізнатися час в секундах, це просто, ділимо epoch на 60 з остачею. Так само як і з кількістю хвилин, якщо число менше 10 то додаємо перед ним 0. Кінцевим результатом після цих розрахунків отримаємо три змінні second, minute та hour. Після конкатенації змінних утвориться змінна realtime яка й буде передаватися на Google Диск.

```
  s.write("s");
  if(s.available()>0)
{
 puls=s.read();

}
if(s.available()>0)
{
 temp=s.read();

}
```

Рис.3.16 Зчитування результатів моніторингу

NodeMCU записує управляючий символ і надсилає його на Arduino Nano та починає прослуховувати данні з нього. Далі за допомогою умовного оператору if та функції available() яка повертає кількість байт(символів) доступних для зчитування з буферу послідовного порту. Під символами розуміються данні, які вже прийняті та зберігаються в послідовному буферу

приймача. Функція available() є наслідувачем допоміжного класу Stream. Отже ми перевіряємо якщо є доступні дані то функція зчитує байт. [18]

Я створюю змінні sheetPulse та sheetTemp в яких будуть дані пульсу та

```
Serial.print("Pulse: ");  Serial.print(puls);
sheetPulse = String(puls);                                    //convert integer h
Serial.print("%  Temperature: ");  Serial.print(temp);  Serial.println("°C ");
sheetTemp = String(temp);
static int error_count = 0;
static int connect_count = 0;
const unsigned int MAX_CONNECT = 20;
static bool flag = false;
payload = payload_base + "\"" + sheetTemp + "," + sheetPulse + "," + realtime + "\"}"
```

Рис.3.17 Ініціалізація змінних для передачі

температури тіла людини. Також я виводжу значення цих змінних на екран. Це знадобиться якщо в процесі рефакторингу коду потрібно буде оцінювати значення змінних. А це швидше робити в додатку Arduino IDE ніж через Google Диск. Також ініціалізується змінна payload в якій через кому буде записано значення змінних sheetTemp, sheetPulse та теперешній час realtime.

Також присутня перевірка, якщо пульс менше можливого у людини то дані на Google Диск відправлятися не будуть. Отже таблиця не буде засмічуватися некоректною інформацію.

```
}if (puls > 40) {
Serial.println("POST or SEND Sensor data to Google Spreadsheet:");
if (client->POST(url2, host, payload)) {
  ;
}
```

Рис.3.18 Перевірка на наявність пульсу

Дані завантажуються за допомогою функції client->Post. Її аргументи це url2 тобто сайт де зберігається скрипт Google Диску. Host це змінна яка зберігає назву сайту. Та payload в якій зберігаються самі дані.

Підсумовуючи функції які виконує програмний код, що завантажується на NodeMCU. Мікроконтролер отримує дві змінні, зі значеннями пульсу та температури, від Arduino Nano. Також відправляє запит на сервер реального часу для отримання часу результатів заміру. Відповідь сервера необхідно

обробити для надання набору чисел читабельної форми у вигляді годин, хвилин та секунд. Отримуючи всі необхідні результати мікроконтролер зв'язується з Google Диском та пересилає інформацію на нього. Далі потрібно налаштувати Google Диск для приймання та відображення даних моніторингу.

Налаштування Google Диску

Google Sheets використовується замість Microsoft Excel для розробки електронних таблиць. Цей інструмент забезпечує хороший спосіб зберігання або обробки даних в формі електронних таблиць та може бути інтегрований з десятками інших сервісів, представлених Google, наприклад Maps. За допомогою Api та використання gscript Google сильно спростив розробниками програмне заповнення даних в Google Sheets. Тим самим вони спрощують створення рішень, що використовують їх сервіс, саме цим я й буду користуватися.

Для відправки даних на Google Sheets, нам необхідно спочатку створити лист, а потім підготувати його для отримання даних з мого пристрою, створивши gscript для прив'язки листа до пристрою. Спочатку потрібно перейти на сайт docs.google.com, доведеться увійти в свій обліковий запис

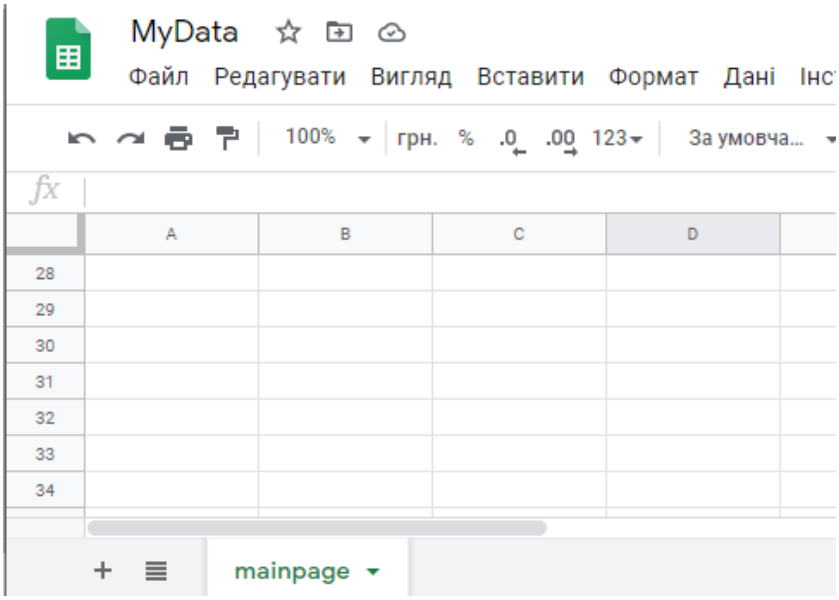


Рис.3.19 Порожня Google таблиця

Google, якщо у вас його немає то потрібно зареєструватися. Коли сторінка відкриється з випадаючого списку потрібно обрати Google sheets.

Після створення електронної таблиці потрібно дати їй ім'я, я назвав "MyData", як видно на рисунку 3.19. Після зміни назви таблиці можна змінити ім'я листа, я назвав "mainpage". Тепер потрібно потрібно створити файл для проекту. В цьому файлі буде написаний код, що буде відповідати за коректну роботу електронної таблиці, та розподіляти дані по стовпцям. Для цього потрібно на панелі інструментів обрати вкладку «Інструменти» та перейти на «Редактор сценаріїв» як показано на рисунку 3.20.

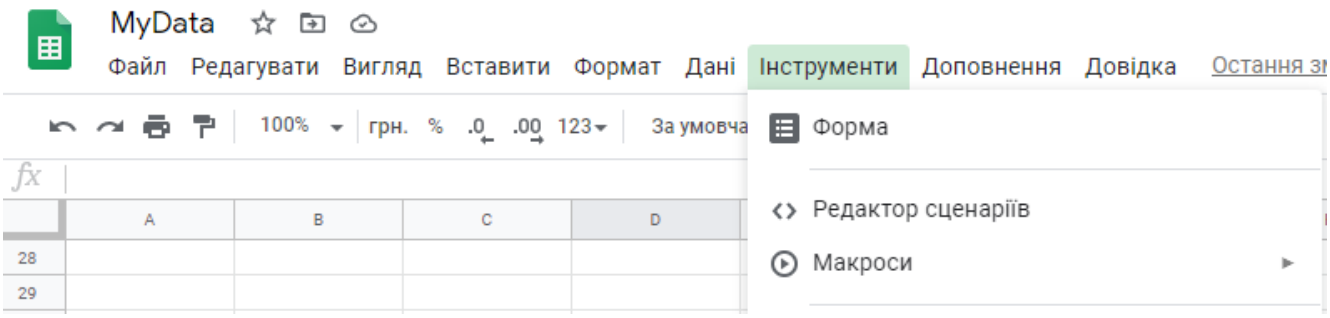


Рис.3.20 Меню створення файлу зі скриптом

Відкриється нова вкладка, в якій буде редактор скриптів Google. Я перейменував цей файл у "Project", для того, щоб він мав зручне для використання ім'я. Далі потрібно написати скрипт який керував би електронною таблицею.[22]

```
var SS = SpreadsheetApp.openById('1nlcfyFKh2nG5ApeOjVAik7WUWGZ8D_fllXp3KuOWcyo');
var sheet = SS.getSheetByName('mainpage');
var str = "";
```

Рис.3.21 Написання коду для Google sheet

На рисунку 3.21 задаються дві змінні. Перша змінна SS це посилання на електронну таблицю. А друга, sheet це назва листа таблиці на який будуть завантажуватися данні. Функція getSheetByName повертає лист з вказаним

іменм. Якщо декілька листів мають одне й те саме ім’я повертається те яке стоїть раніше. Або повертає null якщо немає листа з вказаним іменем.

```
var range = sheet.getRange('A1');
var retval = range.setValue(val).getValue();
var now = Utilities.formatDate(new Date(), "
//sheet.getRange('B1').setValue(now);
sheet.getRange('B1').setValue('BPM');
sheet.getRange('C1').setValue('Time');
```

Рис.3.22 функція getRange()

За допомогою виклику getRange(‘A1’) можна отримати секцію A1 на першому листі. Клас range використовується для доступу та зміни діапазонів електронної таблиці. Діапазон може бути окремою секцією на листі або групою сусідніх секцій на листі. Метод класу setValue() встановлює значення діапазону. Значення може бути числовим, строковим, логічним або датою. Метод getValue() повертає значення верхньої лівої секції в діапазоні. Значення може бути різних типів в залежності від значення секції. Порожні секції повертають порожній рядок. Далі за допомогою функції setValue() встановлюю у секціях B1 та C1 значення BPM та Time, для того щоб було зрозуміло в який стовпець які значення записуються. [23]

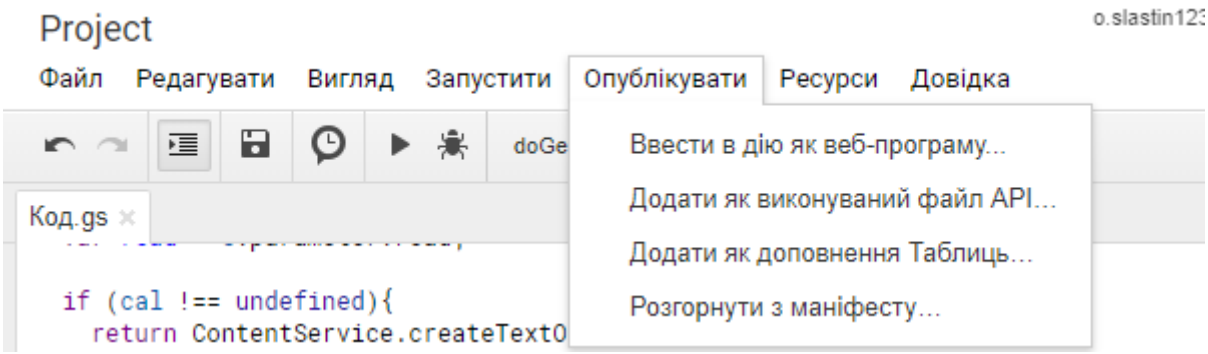


Рис.3.23 Створення веб-програми

Після того як весь програмний код написаний для функціонування електронної таблиці написаний потрібно опублікувати його. Це потрібно для збереження коду та його автономного запуску. Якщо цього не зробити то електрона таблиця просто не зможе приймати дані які ви будете на неї

надсилати. На верхній панелі інструментів вашого проекту варто обрати вкладку «Опублікувати», а у з’явившемуся меню натискаю на кнопку «Ввести в дію як веб-програму» та переходжу до сторінки публікації. Там необхідно заповнити безліч інформації, вказати нову версію проекту, додати e-mail того хто публікую програму, а також написати кому надається доступ до додатку, хто може змінювати програму, та додавати дані. Важливо вказати саме нову версію проекту, оскільки якщо цього не зробити і просто оновити стару версію то програмний код не оновиться і ваша програма залишиться без змін. Після всього потрібно надати кілька дозволів і програма готова до роботи.

ВИСНОВКИ ДО РОЗДІЛУ 3

В цьому розділі виконано моделювання пристрою стеження за станом пацієнта. Використовуючи плату та датчики, вибір яких було детально описано в попередньому розділі, було намальовано схему та по ній створено сам пристрій. Замість використання макетної плати яка б збільшила не тільки розмір девайсу, а й його коштовність, була використано пайка проводів. Також даний метод з'єднання дозволив збільшити міцність проекту.

А використовуючи середовище розробки Arduino IDE, яке було обрано серед інших через те що воно найкраще підходить для даного проекту, був написаний програмний код. Створено два окремі скетчі, один завантажений на Arduino Nano та відповідає за отримання даних моніторингу за датчиків та пересилку цих даних на Wi-Fi модуль. Інший скетч виконує набагато більше функцій. Він окрім того, що отримує дані стану пацієнта, також під'єднується до сервера реального часу, за допомогою NTP протоколу, для отримання часу зняття показників пульсу та температури. Також потім ці дані він записує в змінну та надсилає на Google Диск.

Таким чином в цьому розділі було описано та розроблено сам пристрій, його програмування та налаштування.

РОЗДІЛ 4
ДЕМОНСТРАЦІЯ РОБОТИ ПРИСТРОЮ

4.1. Інструкція користувача

Після того як програмний код написано, допрацьовано та завантажено на обидві плати, а Google Диск налаштовано на приймання даних можна запускати пристрій. Оскільки плата NodeMCU через контакт Vin живиться від 5 вольтового контакту на платі Arduino Nano, то необхідний лише один роз'єм USB для повноцінного функціонування девайсу. Після подачі живлення на плату, незалежно від того під'єднана вона до комп'ютеру чи блоку живлення, через 10 секунд починають вимірюватися температура та пульс людини.

Проте є декілька правил яких необхідно дотримуватися при вимірюванні пульсу:

- Потрібно відпочити 3-5 хвилин, та при цьому не розмовляти, перед тим як проводити вимірювання;
- Сядьте в зручне крісло, що підтримує спину, ноги та руки при цьому тримати не схрещеними;
- Руки необхідно покласти на стіл на рівні серця або на іншу рівню поверхню;

Коли пристрій запущено результати вимірів можна побачити в меню монітор порту в середовищі Arduino IDE. Для того щоб зайти в монітор порту необхідно виконати декілька кроків. Зайти в середовище Arduino IDE та на панелі інструментів обрати вкладку «Інструменти». Натиснувши на неї потрібно обрати порт до якого під'єднано пристрій. Це номер USB роз'єму вашого ноутбуку чи портативного комп'ютеру. Також можна вибрати назву плати яку ви використовуєте. Це потрібно обов'язково зробити інакше Arduino IDE не зможе з'єднатися з нею.

Рис.4.1 Монітор порту

На рисунку 4.1 видно як відображаються результати в моніторі порту. Після того як пульс знайдено виводиться напис “Start pulse”, а потім значення пульсу та температури які оновлюються кожні 8 секунд.

Коли пристрій запуститься, на Google Диску якщо він порожній колонки отримають відповідні назви як це показано на рисунку 4.2. Якщо ж у ньому вже є записи результатів, то нові будуть додаватися знизу колонок до попередніх.

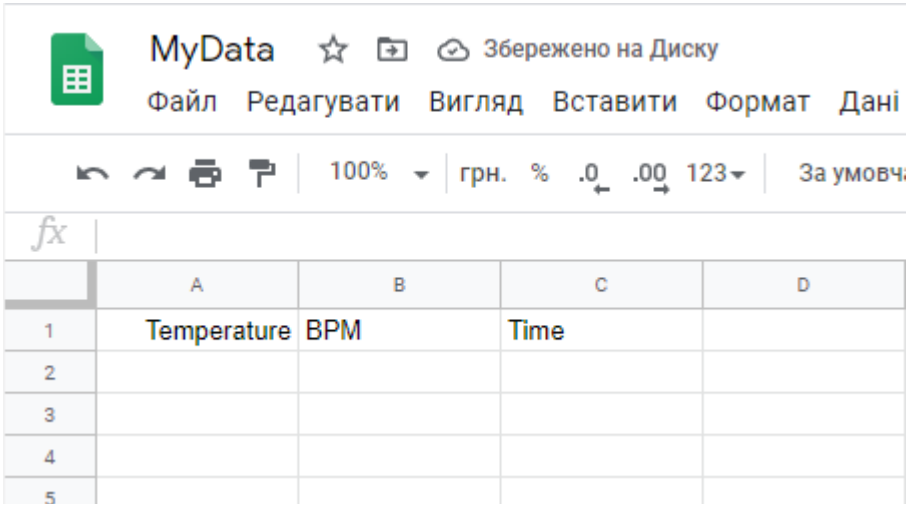
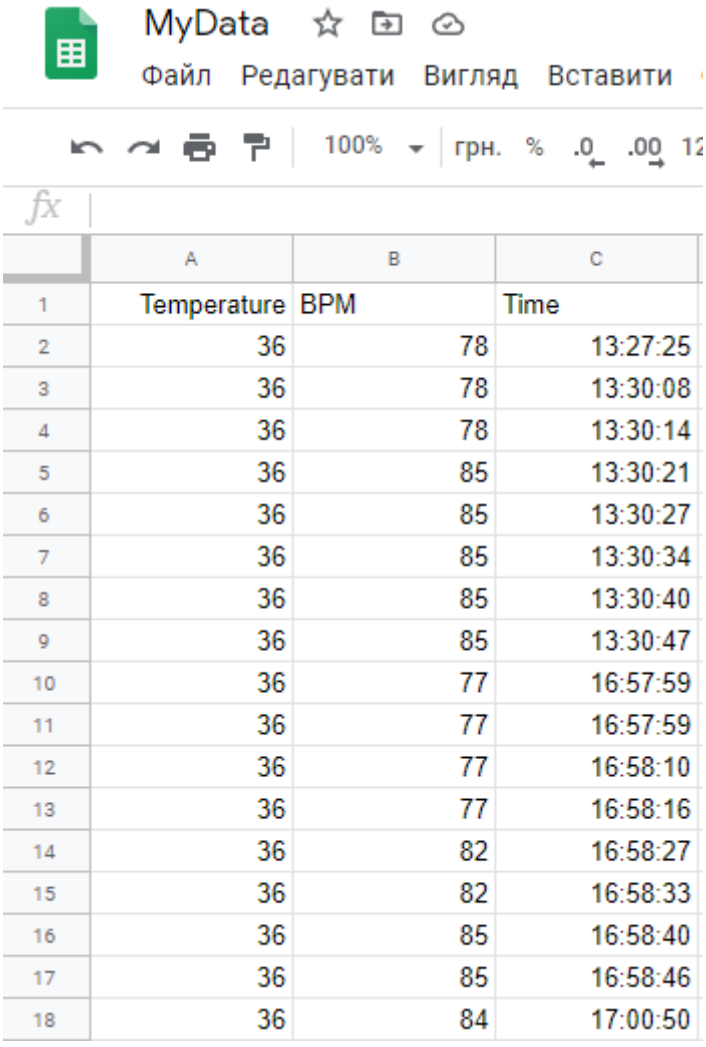


Рис.4.2 Нові колонки в електронній таблиці

Після створення нових колонок в електронній таблиці, вони кожні 8 секунд заповнюються новими даними. На рисунку 4.3 видно заповнену таблицю. У ній присутні дані температури, результати моніторингу пульсу, а також точний час коли цей замір був зроблений.



	A	B	C
1	Temperature	BPM	Time
2	36	78	13:27:25
3	36	78	13:30:08
4	36	78	13:30:14
5	36	85	13:30:21
6	36	85	13:30:27
7	36	85	13:30:34
8	36	85	13:30:40
9	36	85	13:30:47
10	36	77	16:57:59
11	36	77	16:57:59
12	36	77	16:58:10
13	36	77	16:58:16
14	36	82	16:58:27
15	36	82	16:58:33
16	36	85	16:58:40
17	36	85	16:58:46
18	36	84	17:00:50

Рис.4.3 Заповнена електронна таблиця

Цю електронну таблицю можете переглядати ви самі. Або ж лікар чи будь-яка людина якій буде надано доступ. Оскільки датчики не володіють великою точністю виникає незначна похибка, вона особлива помітна на початку моніторингу коли пристрій налаштовується.

ВИСНОВОК ДО РОЗДІЛУ 4

В цьому розділі була описана робота пристрою стеження за станом пацієнта. Він дуже легкий у використанні, та може використовуватися самостійно будь-ким.

Показано та описано інтерфейс користувача, він є зрозумілим і не потребує довгої адаптації. На комп’ютер або телефон не потрібно встановлювати жодних додатків, а отже для користування пристроєм необхідна лише наявність інтернету.

ВИСНОВКИ

В дані бакалаврській роботі була розроблена та проаналізована робота системи стеження за здоров'ям та температурою пацієнта за допомогою плат Arduino. Серцебиття вимірювалося за допомогою імпульсного датчику на основі світлодіоду Pulse Sensor, а температура вимірювалася цифровим датчиком DS18B20. Обидва результати були оброблені за допомогою плати Arduino Nano та за допомогою послідовного інтерфейсу передаються на Wi-Fi модуль NodeMCU. Звідки через мережу Інтернет дані завантажуються на Google Диск.

Був змодельований та розроблений виріб який не потребує доробок для того щоб використовувати його. Через те що у нього немає зовнішньої оболонки він виглядає досить неохайно. Також використані датчики займають більше місця ніж їх більш дорогі та якісні аналоги. Написаний програмний код можна доповнити як і саму модель пристрою, але в цій комплектації він функціонує на достатньому рівні. При заміні датчиків на нові програмний код не потребує великих змін, потрібно буде просто замінити назви датчиків в середовищі розробки.

Для майбутнього використання та підстроювання до тенденцій пристрій можна доробити:

- Можна додати звук, для того щоб пристрій видавав сигнал кожен раз при отриманні імпульсу або вмикав сигнал тривоги при ненормальному стані здоров'я;
- Вивід може бути відправлений на телефон за допомогою GSM або Bluetooth модуля;
- Додати додаткові параметри, наприклад артеріальний тиск;
- Можна під'єднати нейронну мережу яка була б навчана на аналіз стану та самопочуття людини та могла б попереджати якщо дані моніторингу людини виходять за межі допустимих норм;

Отже даний розроблений проєкт є перспективним і його можна вдосконалювати якщо у цьому виникне необхідність. Були виконані всі поставлені до роботи завдання та досягнуті цілі

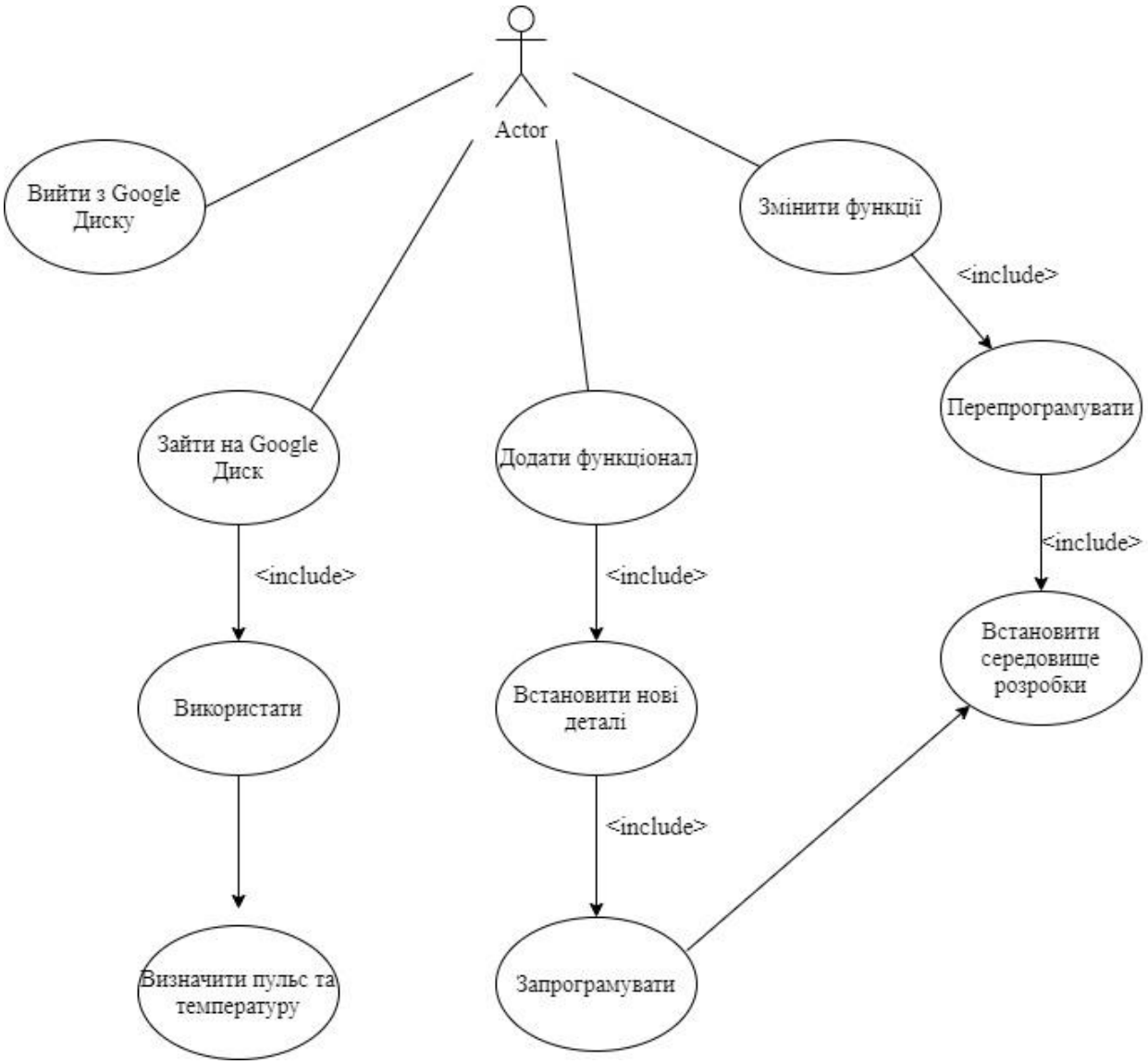
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. http://www.sut.ru/doci/nauka/review/20164/67-80.pdf
2. https://ilounge.ua/products/fitnes-braslet-fitbit-charge-2-large-black-stainless-steel-kupit
3. https://mhealthspot.com/2014/08/wemu-smart-clothing-improves-epilepsy monitoring-diagnosis/
4. https://www.arduino.cc/en/guide/introduction
5. https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-nano.html
6. https://www.arduino.cc/en/Main/Arduino_BoardLeonardo
7. https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf
8. https://lesson.iarduino.ru/page/urok-27-pulsometr/
9. https://3d-diy.ru/wiki/arduino-datchiki/datchik-pulsa/
10. https://www.theengineeringprojects.com/2018/10/introduction-to-nodemcu-v3.html
11. https://myrobot.ru/wiki/index.php?n=Experiences.NodemcuV3Pinout
12. https://soltau.ru/index.php/themes/kompyutery-i-programmy/item/465-kakie-sushchestvuyut-sredy-razrabotki-ide-dlya-arduino
13. http://arduinodev.com/codeblocks/
14. https://www.theengineeringprojects.com/2018/10/introduction-to-arduino-ide.html
15. https://doc.arduino.ua/ru/prog/Include
16. https://doc.arduino.ua/ru/prog/SoftwareSerial
17. https://arduinomaster.ru/program/arduino-void-loop-i-void-setup/
18. https://doc.arduino.ua/ru/prog/SoftwareSerialAvailable
19. https://en.wikipedia.org/wiki/HTTPS
20. https://ru.wikipedia.org/wiki/NTP
21. https://ru.wikipedia.org/wiki/TLS

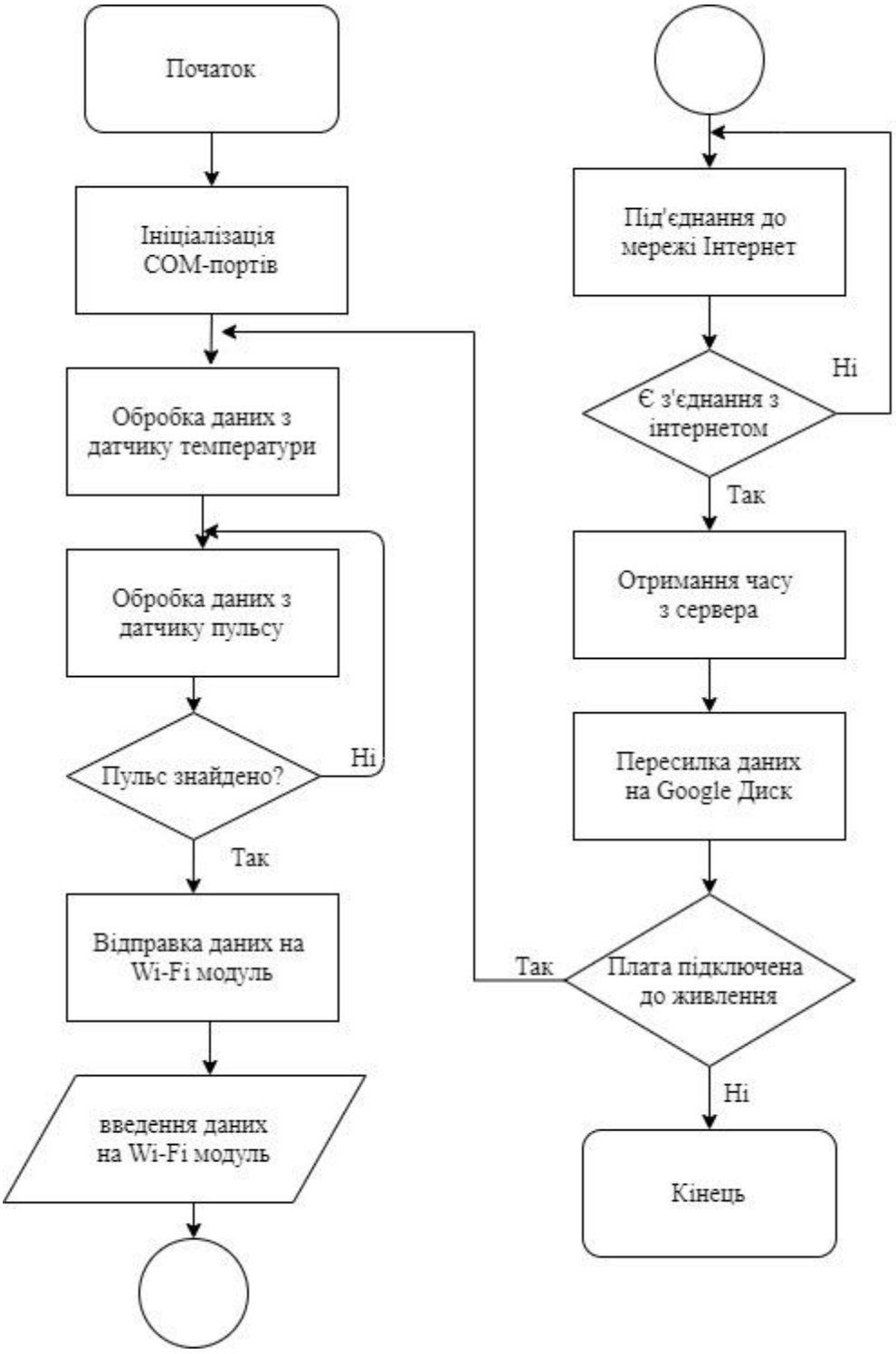
22. https://developers.google.com/sheets
23. https://developers.google.com/apps-script/reference/spreadsheet/sheet

ДОДАТОК А

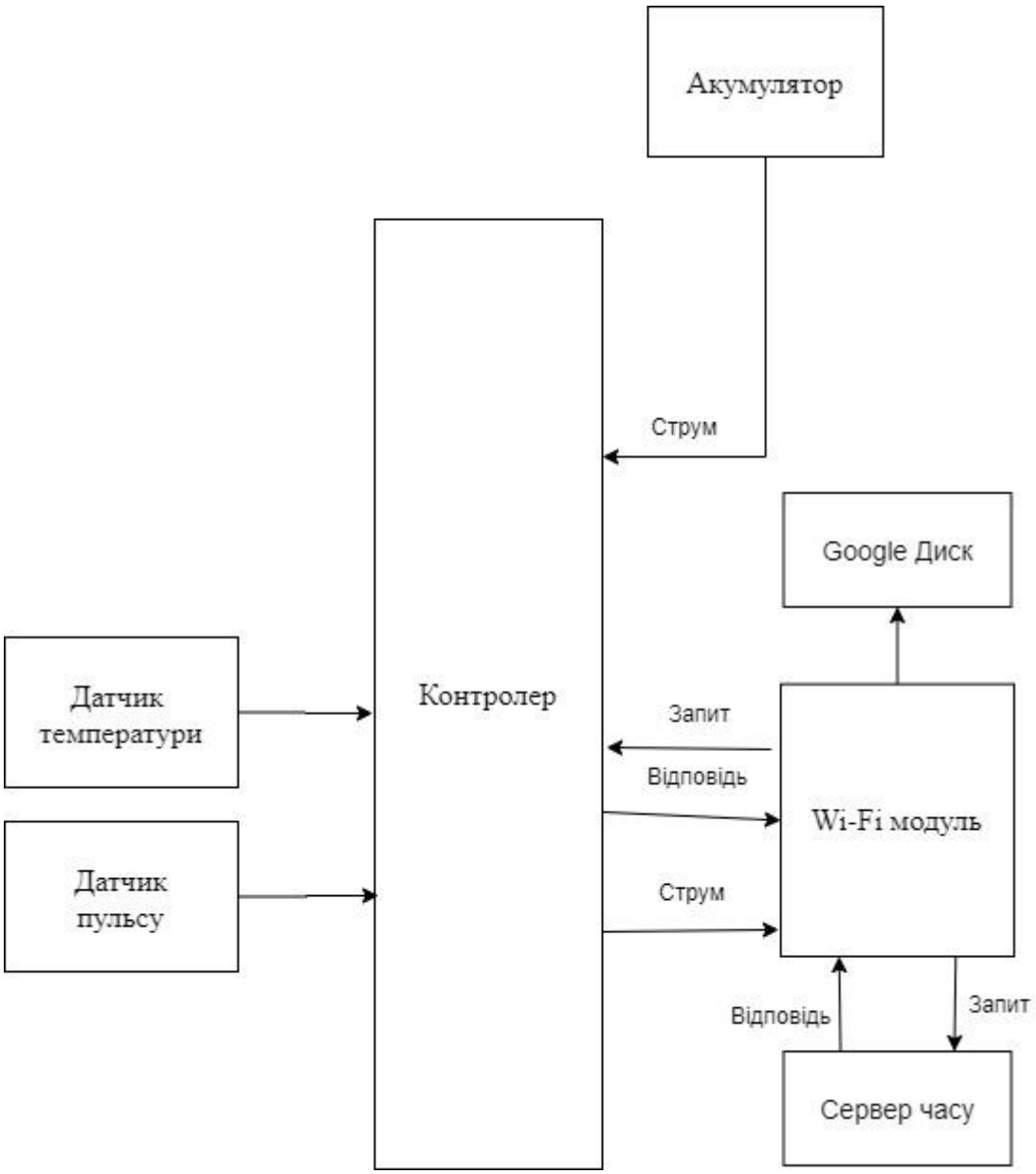
Київ – 2019



					ІАЛЦ.467100.004 Д1	
Зм.	Аркуш	№ докум.	Підп.	Дата		
Розроб.		Сластін О.Ю.			Система стеження за станом пацієнта Структурна схема сценарію використання	Літ. / Аркуш / Аркушів
Перевір.		Сімоненко В.П.				НТУУ(КПІ) ФІОТ група ІО-63
Н. контр.		Сімоненко В.П.				
Затв.						



					ІАЛЦ.467100.005 Д2	
Зм.	Аркуш	№ докум.	Підп.	Дата		
Розроб.		Сластін О.Ю.			Система стеження за станом пацієнта Блок-схема алгоритму	Літ. / Аркуш / Аркушів
Перевір.		Сімоненко В.П.				
Н. контр.		Сімоненко В.П.				НТУУ(КПІ) ФІОТ група ІО-63
Затв.						



					ІАЛЦ.467100.006 Д3		
Зм.	Аркуш	№ докум.	Підп.	Дата			
Розроб.		Сластін О.Ю.			Система стеження за станом пацієнта Функціональна схема алгоритму	Літ.	Аркуш	Аркушів
Перевір.		Сімоненко В.П.						
						НТУУ(КПІ) ФІОТ група ІО-63		
Н. контр.		Сімоненко В.П.						
Затв.								